(2)

NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A246 190

DTIC
ELECTE
FEB 21 1992
S D
D

THESIS

PROBABILITY MODELS FOR DEFENSE

AGAINST MISSILE ATTACKS

by

Rui Almeida

September, 1991

Thesis Advisor: Donald P. Gaver

Approved for public release; distribution is unlimited

92-03658

92 2 10

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b. OFFICE SYMBOL *(If applicable)* OR	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS *(City, State, and ZIP Code)* Monterey, CA 93943-5000	7b. ADDRESS *(City, State, and ZIP Code)* Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS *(City, State, and ZIP Code)*	10. SOURCE OF FUNDING NUMBERS Program Element No. / Project No / Task No / Work Unit Accession Number

11. TITLE *(Include Security Classification)*
PROBABILITY MODELS FOR DEFENSE AGAINST MISSILE ATTACKS

12. PERSONAL AUTHOR(S) Rui Almeida

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(year, month, day)*	15. PAGE COUNT
Master's Thesis	From To	1991, September	76

16. SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17. COSATI CODES			18. SUBJECT TERMS *(continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUBGROUP	Simulation, Missile Attack, Invisible Kill, Threshold Strategy, Shortest First Rule, Perfect Task Information

19. ABSTRACT *(continue on reverse if necessary and identify by block number)*

Three probability models for defense against mass missile attacks are developed. Each model corresponds to a different level of information. Different strategies are made possible by the information available. The first strategy, Invisible Kill, assigns equal times to the destruction of each missile that is selected for attention. The second, a Threshold strategy, assigns a maximum threshold time to the same task. The last, based on Perfect Task Information, engages missiles in ascending order of their destruction times; the order being assumed known in advance. The expectation of the number of missiles killed is the Measure of Effectiveness used to evaluate the models. Numerical results are analysed through simulation. Different strategies are compared showing the effects of information in defense effectiveness.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
X UCLASSIFIED/UNLIMITED ☐ SAME AS REPORT ☐ DTIC USERS	Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Donald P. Gaver	22b. TELEPHONE *(Include Area code)* (408) 646-2605	22c. OFFICE SYMBOL OR/Gv

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

Approved for public release; distribution is unlimited.

Probability Models
for
Defense Against Missile Attacks

by

Rui Almeida
Lieutenant, Portuguese Navy
B.S., Portuguese Naval Academy , 1983

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September, 1991

Author: Rui M. Prata de Almeida
Rui Almeida

Approved by: Donald P. Gaver
Donald P. Gaver, Thesis Advisor

Patricia A. Jacobs
Patricia A. Jacobs, Second Reader

Peter Purdue
Peter Purdue, Chairman
Department of Operations Research

ABSTRACT

Three probability models for defense against mass missile attacks are developed. Each model corresponds to a different level of information. Different strategies are made possible by the information available. The first strategy, Invisible Kill, assigns equal times to the destruction of each missile that is selected for attention. The second, a Threshold strategy, assigns a maximum threshold time to the same task. The last, based on Perfect Task Information, engages missiles in ascending order of their destruction times; the order being assumed known in advance. The Expectation of the number of missiles killed is the Measure of Effectiveness used to evaluate the models. Numerical results are analyzed through simulation. Different strategies are compared showing the effects of information in defense effectiveness.

Accesion For	
NTIS CRA&I	☑
DTIC TAB	☐
U announced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. INTRODUCTION

Information is known to be an important factor in the determination of the outcome of military combat. Consequently, the addition of information-gathering and interpretation assets must be viewed as in legitimate competition with the acquisition of weapons. In fact, modern "smart" weapons possess within themselves a combination of brawn: destructive power (e.g. high explosive warhead); and brain: navigation, identification, guidance and homing power (e.g. on-board sensors and logic). External C^2 systems are needed to guide overall decision-making: on the defender's side to detect attacks and to schedule defense response; on the attacker's side to first soften or confuse (EW measures, deception) the defender prior to a focused destructive action against sensitive assets.

This thesis seeks to investigate the effects and value of information availability in a defense against a single mass (simultaneous) missile attack. The analysis is carried out from the point of view of a defender operating within constraints of a fixed time window, presumably until destruction of the attacking missiles is no longer possible at the level considered; "leaking" missiles may be dealt with at an ensuing defense level. For simplicity we postulate that the objective of the defender is to maximize the expected number of missiles destroyed at the level under consideration. The defender is using some weapon system (interceptor missiles, irradiation weapons,...). This generic weapon system is characterized by its probability of acquiring a target, acquisition or setup time, and rate of kill. These parameters are supposedly design features intrinsic to the system. It is possible that trade-offs between the weapon system characteristics (parameters) exist, in which

case our models may provide some insight in deciding the desirable parameter configuration.

The three models we develop are of a probabilistic nature, each of them corresponding to a different information level available to the defender. A higher information level should enable the user to enhance his strategy and consequently achieve a gain in defense effectiveness. The general methodology followed is to start with the basic assumptions of the model and strategy, and work our way analytically to explicit expressions for the chosen measures of effectiveness. Numerical results of the probability models are checked against those of discrete-event simulations.

It must be noted that, in this context, the term "information" is used in a specific but informal sense appropriate to the present application. There is no direct connection with such classical concepts as Shannon information or Fisher information. Finally, note that the present problem may appear in other guises, in which a number of tasks must be performed under time constraints; e.g., in repair and maintenance or flexible manufacturing.

II. MASS ATTACK WITH INVISIBLE KILL

A. INTRODUCTION

This chapter deals with the following situation: A defender is under a mass or simultaneous attack from a possibly large number of incoming missiles (a "cluster"). Available to the defender is an unspecified outer-layer weapon system, D1, which enables him to shoot at least some of the missiles. Each missile, upon being detected must be acquired, and only then can one attempt to shoot it down. The weapon system only allows the defender to engage one target at a time, so he faces the problem of how to optimally assign time to each missile so as to maximize some measure of the number of missiles destroyed. It is understood that one is working in a restricted time window: if not destroyed, the missiles will proceed towards their targets, passing through a secondary defence level. The missiles that are not killed by D1 are said to leak to the next layer. The general objective is to minimize leakage.

B. ASSUMPTIONS

In the probability model the following simplifying assumptions will be used:

- The number of attacking missiles is known.
- The probability of acquiring each target is known and fixed; range dependence is ignored.
- Given that the missile is acquired, the probability distribution of the time to destroy it is known.
- It is not known to the defender whether a given missile was successfully acquired or, if acquired was actually killed. This is the minimal information situation; we call it **Invisible Kill.**

- There is a fixed setup time expended each time a new attempt is made to acquire another missile.

Finally, the strategy of the defender will be to allocate time evenly between those missiles selected for attention by D1. Not all attacking missiles will be chosen under optimal conditions; some will be allowed to leak, presumably with warning transmitted to the second (inner) defence layer.

C. MODEL

The following notation will be used: Let n be the number of incoming missiles. The probability of acquiring a target is p , and $q = 1-p$ the probability of acquisition failure. Denote by X the random time to destroy a missile which has been acquired. T is the time available for destroying the targets - the duration of the window of opportunity. There is a fixed time necessary for each acquisition which will be denoted by w.

It is assumed that the distribution of X is known. Call its CDF $F_X(x)$, where:

$$X \sim F_X(x) = \textit{Prob}\{ X \le x \mid \textit{Target was acquired} \} . \qquad (2.1)$$

Suppose one chooses to engage a number, m, of the incoming missiles; $m \le n$. Recalling the declared strategy of assigning equal time to each one, then $T = mw + mx$; here x is the time allowed to destroy a single missile. It follows that:

$$x = \frac{T}{m} - w \, . \tag{2.2}$$

Since each target must be destroyed in less than time x, Prob{destroying a target} = Prob{a target is acquired upon attempt} * Prob{a target is destroyed in the time available: $\max(0, T/m - w)$ }; i.e.:

$$Prob\{destroying\ a\ target\} = p \, . \, F_X\left(\frac{T}{m} - w\right) . \tag{2.3}$$

Note that since $X \geq 0$ with probability one, automatically $F_X(x) = 0$ if $x \leq 0$.

If it is assumed that each attempt to acquire and destroy a target is an independent *Bernoulli* trial, then the number of targets destroyed out of m, $N(T)$, will be distributed

$$N(T) \sim Binomial(m, pF_X(\frac{T}{m} - w)), \tag{2.4}$$

and the expected number of targets destroyed will be

$$E[N(T)] = m p F_X(\frac{T}{m} - w) . \tag{2.5}$$

The current problem is to find the optimal value of m, i.e., one that maximizes this expected value. Other criteria and controls are of course possible. Note that if the targets are known to be <u>different</u>, then a strategy that schedules different times to each type would be reasonable. An extension of this formulation might include a preliminary classification of target type and subsequent allocation of effort to the most profitable types first.

D. MODEL DEVELOPMENT

The optimal value of m will now be derived: namely the number of missiles the defender should engage in order to destroy the maximal expected number of them at his particular defense layer.

The expected number of destroyed or killed missiles is given by (2.5). We wish to find the maximum of this function. Set the derivative $\frac{\partial E[N(T)]}{\partial m} = 0$, implying that if the optimal value, $m = m^o$, is interior, $0 < m^o < n$, then

$$F_X(z - w) = z f_X(z - w), \tag{2.6}$$

where the parameter $z = T/m^o$. In principle there can be several such stationary points, and furthermore the optimal number can be a boundary value: $m^o = n$.

In particular suppose $X \sim Exponential(\lambda)$, where the rate parameter may be interpreted as the number of acquired missiles the weapon kills per unit time. Then:

$$F_X(x) = 1 - e^{-\lambda x}, \quad x > 0 ; \tag{2.7}$$

The expected number of destroyed missiles becomes:

$$E[N(T)] = mp[1 - e^{-\lambda(\frac{T}{m} - w)}]. \tag{2.8}$$

To maximize this function of m, take the derivative and set it equal to zero. After some algebra, get:

$$\ln(m) - \ln(m + \lambda T) + \frac{\lambda T}{m} - \lambda w = 0 ; \tag{2.9}$$

An approximate explicit solution for m^*, a stationary point, even if somewhat inaccurate, would be desirable to show its dependency on the various parameters. Start by rewriting (2.9), and using a Taylor expansion:

$$\frac{\lambda T}{m} - \lambda w = \ln(m+\lambda T) - \ln m$$

$$= \ln(1 + \frac{\lambda T}{m}) \approx \frac{\lambda T}{m} - \frac{(\lambda T)^2}{2m^2}, \tag{2.10}$$

solving for m, yields the approximate explicit result:

$$m^* \approx T\sqrt{\frac{\lambda}{2w}}. \tag{2.11}$$

This number will be a positive real. We want an integer solution; therefore the greatest integer not exceeding m^*, $[m^*]$, and the next lowest integer, $<m^*>$, must be checked to see which one has the largest objective function (2.8). Call this number m^{**}, then the approximate optimal m^o will be:

$$m^0 = \min(n, m^{**}). \tag{2.12}$$

Accordingly the corresponding optimal value of the expected number of missiles killed is, again, approximately:

$$E^o[N(T)] = m^o p\,[1 - e^{-\lambda\left(\frac{T}{m^o} - w\right)}]. \tag{2.13}$$

Note that the second derivative of $E[N(T)]$ is

$$\frac{\partial^2 E[N(T)]}{\partial m^2} = -p\,\frac{(\lambda T)^2}{m^3}\,e^{-\lambda(\frac{T}{m} - w)}; \tag{2.14}$$

therefore

$$\frac{\partial^2 E[N(T)]}{\partial m^2} < 0, \ \forall m . \qquad (2.15)$$

The function $E[N(T)]$ is concave throughout the domain of m. In this case m^o, if obtained as described above, will be the approximate unique global maximizer.

E. SIMULATION

To verify the analytical results obtained thus far, a terminating simulation [Ref. 1] was performed. The measure of effectiveness estimated is the Expected Number of Missiles Killed per Attack. The attacks were replicated independently, 15000 times, with several combinations of input parameters. The FORTRAN codes pertaining to the Event routines are displayed in Appendix A. For Timing and Calendar manipulation routines the SIMUTIL package [Ref. 2] was employed, as was the LLRANDOMII Random Number Generator [Ref. 3].

The results of the simulation were written to CMS files, and used to generate several plots of response surfaces. One of these plots is shown in Figure 2.1, with E[Missiles Killed] as the response variable.

For example, if one takes the time window to be 700, 20 attacking missiles, the rate of kill 0.01, the acquisition time 10 and the probability of acquisition 0.85; then solving (2.11) yields $m^* = 15.652476$. Checking $m^* = 15, 16$ with (2.13) reveals that $m^{**} = 15$; therefore $m^o = \min(15, 20) = 15$. It can be observed from Figure 2.1, with these values, that the approximate analytical optimal seems to be in good accordance with the simulation results. Note that obtaining the precisely correct value of m is not important; the value of the expected number killed is not very sensitive to that value.

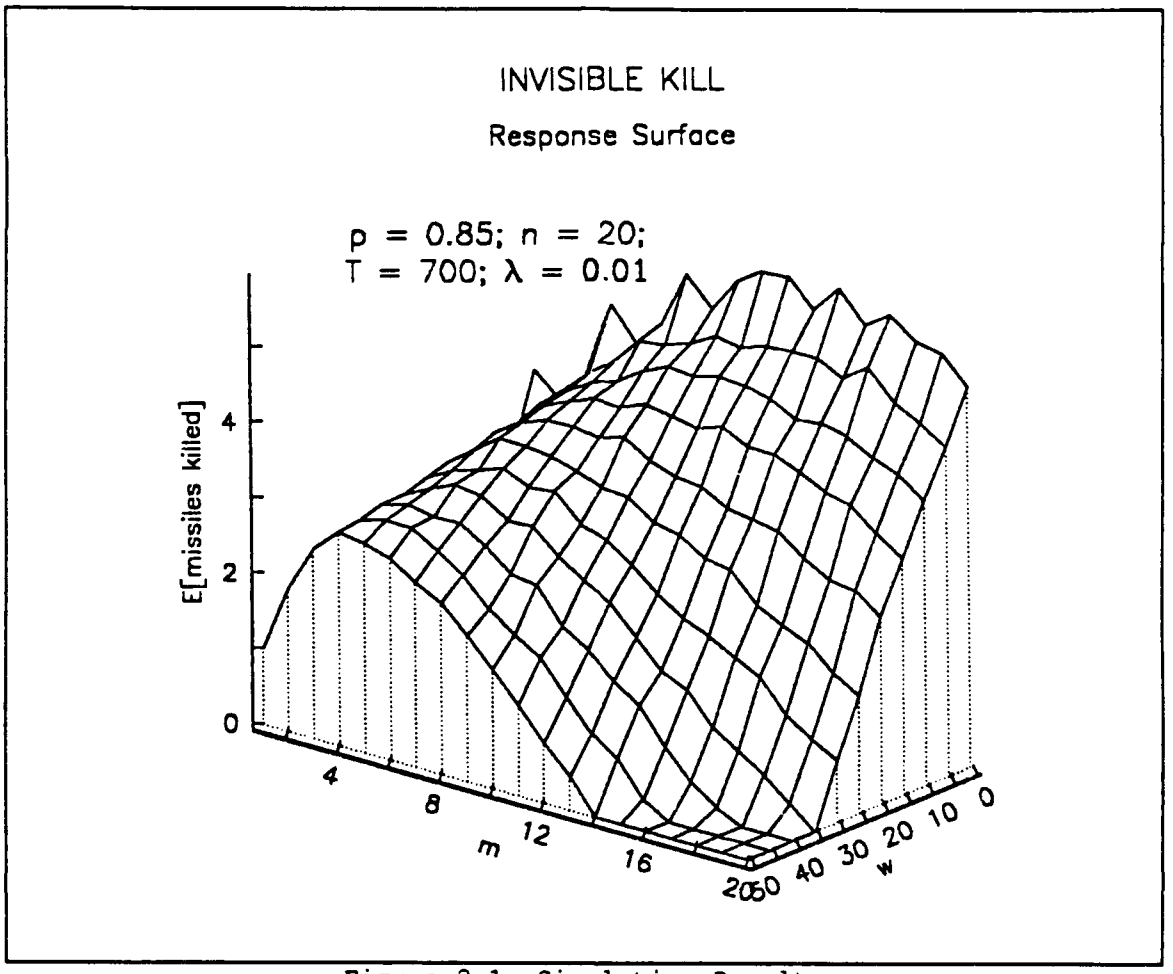

Figure 2.1 Simulation Results

F. USE OF DECOYS BY ATTACKER

It might be of some practical interest to discuss the influence of the use of decoys, in the results of the previous strategy for Invisible Kill. By assumption the decoys will be randomly mixed with the missiles within the cluster. As information available is minimal, the decoys will be undistinguishable from the missiles for the defender; as a result the defender employs the same strategy regardless of their presence.

Let r be the fraction of targets which are true missiles, i.e.,

$$r = \frac{number\ of\ missiles}{number\ of\ missiles\ +\ number\ of\ decoys}\ . \qquad (2.16)$$

Using the current independence assumptions, the number of missiles destroyed in m trials will be distributed *Binomial* and its mean will be:

$$E[missiles\ destroyed] = mprF_X(\frac{T}{m} - w)\ . \qquad (2.17)$$

If the destruction time is again taken to be *Exponential*, the same steps already taken show that the optimal number of targets to engage does not vary with r. The expected number of missiles destroyed using the optimal strategy decreases, being multiplied by a factor of r. This result is rather intuitive. Since there is no way the defender can assess the composition of the attacking target cluster, defense effectiveness depends directly on the fraction of decoys. Eventually an attack with large numbers of decoys will succeed in diluting the defender's effort, thus achieving a high rate of penetration to a subsequent defense layer.

It is interesting to conjecture the payoff from a subsystem capable of classifying attacking entities as either missiles or decoys with some probability ("skill"). It may be that tradeoff between some of the other parameters, e.g., T, w, or λ, and classification skill can be beneficial for the overall system effectiveness. This tradeoff is susceptible to quantitative study conducted in the mode described throughout this thesis.

III. VISIBLE KILL - A THRESHOLD STRATEGY

A. INTRODUCTION

The former chapter dealt with a situation in which there is no direct information pertaining to when, or if, a given target is destroyed. The previous assumptions are next modified in a way such that the defender recognizes the instant when a target is destroyed (**VISIBLE KILL**), if such an event occurs. Upon destruction of a target the defender immediately attempts to acquire another (if any are left). It should be noted that acquisition is still 'not visible', i.e. after the fixed acquisition time w, the defender does not know if the acquisition was a success. Thus D1 may have to wait indefinitely for news that a kill has occurred, which leads to consideration of a time threshold strategy or policy.

B. THRESHOLD MODEL

A threshold policy is adopted: the defender will allow a maximum threshold time , τ, to be allocated to destruction of a target. If, after time τ elapses, the target is not observed to be destroyed the defender will go back to acquisition mode. Such a policy is easy to implement and can be justified by dynamic programming under certain circumstances.

Define $Z(\tau)$, the total random time to kill one target with a threshold equal to τ. Then, from the assumptions made, the following recursion must hold:

$$Z(\tau)\,|\,X=x \;=\; \begin{cases} w+x & \text{if } x \le \tau \\ w+\tau+Z'(\tau) & \text{if } x>\tau \end{cases}, \tag{3.1}$$

where X is the random time to destroy an acquired target, in a continuous engagement. $Z'(\tau)$ is a random variable having the same distribution as $Z(\tau)$; it is the time to shoot down a target if the first attempt fails and the process of acquisition must start over. The CDF of X is known to be $F_X(x)$.

The upper branch of (3.1) occurs when acquisition is successful and the target is shot within the prescribed threshold. This has probability $p\,dF_X(x)$. The lower branch occurs if either acquisition fails, or acquisition succeeds but the missile is not shot in the allowed threshold time. This has probability $q + p\bar{F}_X(\tau)$. Thus we obtain the unconditional expectation:

$$E[Z(\tau)] \;=\; \frac{w + \int_0^{\tau} p x\, dF_X(x) \;+ \tau\,[q + p\bar{F}_X(\tau)]}{p\,F_X(\tau)}; \tag{3.2}$$

where $\bar{F}_X(x) = 1 - F_X(x)$. Integration by parts in the numerator gives this slightly simpler form:

$$E[Z(\tau)] \;=\; \frac{w + p\int_0^{\tau} \bar{F}_X(x)\,dx \;+ \tau q}{p\,F_X(\tau)}. \tag{3.3}$$

The defender wishes to maximize $E[N(T)]$, with τ being the decision variable. Therefore we must express $E[N(T)]$ in terms of the expectation (3.3) just derived.

If we interpret each missile destruction as a point event, then $Z(\tau)$ would be the random interarrival time. If we had a "continuous supply" of missiles, and a large time window , T, then from the elementary renewal theorem [Ref. 4]:

$$E[N(T)] \sim \frac{T}{E[Z(\tau)]}. \tag{3.4}$$

In fact, a correction term might be added [Ref. 4]:

$$E[N(T)] \sim \frac{T}{E[Z(\tau)]} + \frac{Var[Z(\tau)] - E^2[Z(\tau)]}{2E^2[Z(\tau)]}. \tag{3.5}$$

This requires the computation of the variance of $Z(\tau)$ which is more difficult and leads to a complex expression. In this paper we retain the approximation (3.4), accepting the consequent loss of accuracy. Thus we use

$$E[N(T)] \approx \frac{Tp\,F_X(\tau)}{w + p\int_0^{\tau} \overline{F}_X(x)dx + \tau q}. \tag{3.6}$$

The defender's problem is to find:

$$\tau^o = \underset{0 < \tau \le T}{argmax}\, E[N(T)] \tag{3.7}$$

τ^o, the optimal threshold can be looked for among the critical points on the interior of its domain, or at $\tau = T$.

For example, say $X \sim Exponential(\lambda)$. After substitution of the appropriate expressions for the distribution:

$$E[N(T)] \approx \frac{Tp(1-e^{-\lambda\tau})}{w+\frac{p}{\lambda}(1-e^{-\lambda\tau})+\tau q}; \tag{3.8}$$

Simulations were conducted to study the dependence of $E[N(T)]$ on the parameters λ, w, τ, p. Plots resulting from these simulations are displayed and discussed on Appendix B.

Again we seek the critical points of $E[N(T)]$ by finding the zeros of its derivative:

$$\frac{\partial E[N(T)]}{\partial \tau} = 0 \rightarrow (\lambda\tau+\frac{\lambda w}{q}+1)e^{-\lambda\tau} = 1, \tag{3.9}$$

or equivalently, by taking logarithms of both sides:

$$\ln(\lambda\tau+\frac{\lambda w}{q}+1) = \lambda\tau. \tag{3.10}$$

If $p = 1$, it can be verified that $\tau^o = T$, otherwise the optimal threshold is to be found as a solution to (3.9) or (3.10).

Two approximate solutions for (3.9) are provided. The first is obtained by rewriting (3.9) as

$$e^{\lambda\tau} - 1 = \lambda\tau + \frac{\lambda w}{q}, \tag{3.11}$$

and expanding the exponential term:

$$e^{\lambda\tau} \approx 1 + \lambda\tau + \frac{(\lambda\tau)^2}{2}. \tag{3.12}$$

Substituting (3.12) into (3.11) and solving for τ yields:

$$\tau^o \approx \sqrt{\frac{2w}{\lambda q}}. \tag{3.13}$$

A second, more accurate solution, is derived by taking (3.13) as the initial guess and performing one iteration of Newton's method to solve (3.11), resulting in:

$$\tau^o \approx \frac{\left(\sqrt{\frac{2w\lambda}{q}} - 1\right) e^{\sqrt{\frac{2w\lambda}{q}}} + \frac{\lambda w}{q} + 1}{\lambda\left(e^{\sqrt{\frac{2w\lambda}{q}}} - 1\right)}. \tag{3.14}$$

Optimal threshold values obtained with the approximations (3.13) and (3.14) are shown in Table 3.1, together with numerical solution of (3.10).

Table 3.1 OPTIMAL THRESHOLD APPROXIMATIONS (p=0.85)

w \ λ		0.01	0.05	0.1
5.0	(3.10)	71.895584	28.074877	18.164104
	(3.13)	81.649658	36.514837	25.819889
	(3.14)	72.721021	29.928094	20.659227
10.0	(3.10)	96.921939	36.328208	22.991621
	(3.13)	115.470054	51.639778	36.514837
	(3.14)	99.285331	41.318454	29.264013
20.0	(3.10)	128.640050	45.983242	28.435654
	(3.13)	163.299316	73.029674	51.639778
	(3.14)	135.311964	58.528026	42.703706

Although the approximations are not always accurate, they are still useful since the expected number of missiles killed is not very sensitive to the threshold value (see Appendix B). Table 3.2 shows values of $E[N(T)]$, obtained from (3.8) with

optimal thresholds given by (3.10), (3.13) and (3.14). As it is seen, the three methods of deriving the optimal threshold produced very similar results.

Table 3.2 EXPECTATION OF MISSILES KILLED (T=700, p=0.85)

w \ λ		0.01	0.05	0.1
5.0	(3.10)	5.138864	20.368877	33.569134
	(3.13)	5.133375	20.176869	32.874955
	(3.14)	5.138821	20.357661	33.476053
10.0	(3.10)	4.777647	16.784567	25.374494
	(3.13)	4.764086	16.437478	24.395642
	(3.14)	4.777394	16.738027	25.100826
20.0	(3.10)	4.271479	12.687247	17.364481
	(3.13)	4.241009	12.197821	16.343659
	(3.14)	4.270129	12.550413	16.865152

C. SIMULATION

Optimal values, $E^{o}[N(T)]$, obtained from (3.8) and (3.10), were compared to those of simulations, for several values of λ. The simulation code is very similar to the program "Threshold Simulation" given in Appendix A. The results are displayed in Figure 3.1.

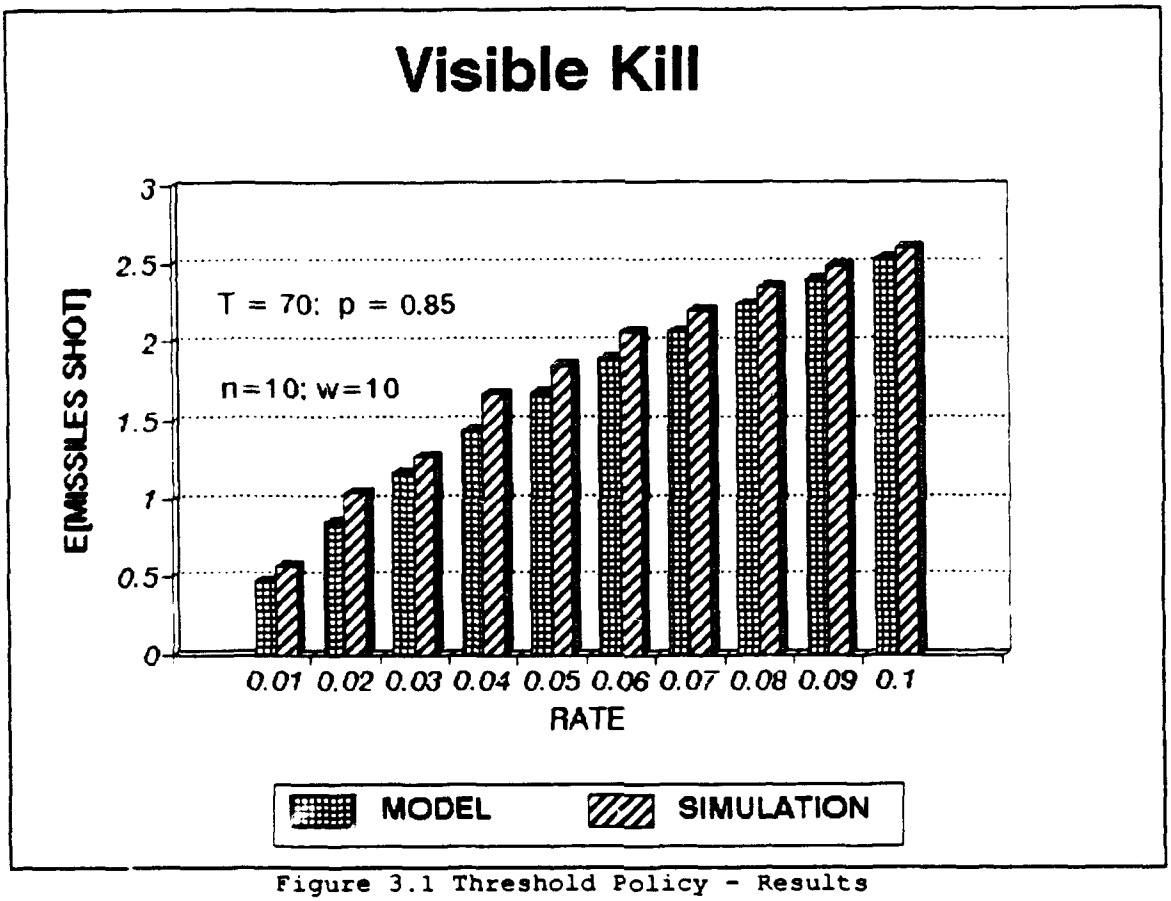

Figure 3.1 Threshold Policy - Results

As can be seen, the model slightly underestimates the simulated values, although its adequacy is seen to improve for larger values of the killing rate.

IV. DEFENSE WITH PERFECT TASK INFORMATION

A. PROBLEM STATEMENT

Suppose one is given n tasks, with the objective of completing the maximum possible number of them within a fixed time window of length T. (E.g., a batch of jobs arriving at a repair facility at the beginning of the day; a cluster of missiles that must be destroyed in limited time). Assume, as an example, that the duration of each task is **known**, and drawn from an *Exponential* distribution. Since the objective is to maximize the number of tasks completed, one obvious strategy is to process the tasks in ascending order of their completion times. This policy will be referred to as a Smallest-First (SF) rule, in the same fashion Coffman [Ref. 5] uses Largest-First. In the context of our missile defense problem the above formulation is artificial because task times would never be known in advance. Note, though, that application of the rule only requires knowledge of the *order*, and not the actual task (shoot-down) time. Therefore the results provide an upper bound on the effectiveness of the outer defense layer, D1.

Denote by X_i the time for completion of the i^{th} task. These times are independent and identically distributed exponentially with parameter λ, their i^{th} order statistic being $X_{(i)}$. If the SF rule is employed, then for some $j < n$:

$$X_{(1)} + \ldots + X_{(j)} \leq T < X_{(1)} + \ldots + X_{(j)} + X_{(j+1)}, \qquad (4.1)$$

in which case only j tasks may be completed or "packed" in time T. Call the number of tasks packed with a SF rule $N(T)$, a random variable. The aim of this chapter is to find the probability mass function (PMF) of $N(T)$, or accurate

approximations to that PMF. This can then be used to evaluate measures of system effectiveness, which can be compared to the results of applying other rules that are based on less information.

Similar problems have been treated in recent literature, although in a more general way and with different objectives and methodology than that presented here. For example, Coffman, Fayolle, Jacquet and Robert [Ref. 5] obtained asymptotic results for the expected number packed, under a Largest-First (LF) rule; with the samples being drawn from a $U(0,1)$ distribution. Coffman, Flatto and Weber [Ref. 6] derived the asymptotic value of the expected number of intervals selected with an optimal 'threshold' rule, for a class of distributions. Theirs is an "ON-LINE" rule, i.e., according to their own definition, the decision to pack or not the interval (job) is made at the time of inspection. The missile problem, i.e. of rank-ordering task times, would also necessarily have to be solved "ON-LINE".

B. EXACT PMF OF N(T)

The following well-known results are relevant to the derivation of the PMF of $N(T)$.

1. Distribution of Spacings

The distribution of the differences between exponential order statistics is next presented. More detail can be found, e.g., in Barlow and Proschan [Ref. 7].

Let $X_1, \ldots, X_n$ be *i.i.d. exponential* with parameter λ. Define the **spacings**, the differences between successive order statistics, as:

$$D_k = X_{(k)} - X_{(k-1)}, \quad with\ X_{(0)} = 0; \qquad k = 1, \ldots, n. \tag{4.2}$$

The normalized spacings $nD_1, (n-1)D_2, \ldots, D_n$ are again *i.i.d. exponential* with rate λ. Let these normalized spacings be denoted by $Y_1, \ldots, Y_n$, respectively. They can be expressed in terms of the order statistics in the following way:

$$\begin{aligned} Y_1 &= nX_{(1)} \\ Y_2 &= (n-1)X_{(2)} - (n-1)X_{(1)} \\ &\vdots \\ Y_j &= (n-j+1)X_{(j)} - (n-j+1)X_{(j-1)}. \end{aligned} \tag{4.3}$$

The last equations, if solved for the order statistics, yield:

$$\begin{aligned} X_{(1)} &= \frac{1}{n}Y_1 \\ X_{(2)} &= \frac{1}{n}Y_1 + \frac{1}{n-1}Y_2 \\ &\vdots \\ X_{(j)} &= \frac{1}{n}Y_1 + \frac{1}{n-1}Y_2 + \ldots + \frac{1}{n-j+1}Y_j. \end{aligned} \tag{4.4}$$

Define the summation of the order statistics:

$$S_j = \sum_{i=1}^{j} X_{(i)} = \sum_{i=1}^{j} \frac{j-i+1}{n-i+1} Y_i, \qquad j = 1, \ldots, n. \tag{4.5}$$

Taking the expectation and variance of the right-hand side of (4.5) produces:

$$E[S_j] = \frac{1}{\lambda}\sum_{i=1}^{j}\frac{j-i+1}{n-i+1}\ ;$$
$$Var[S_j] = \frac{1}{\lambda^2}\sum_{i=1}^{j}\left(\frac{j-i+1}{n-i+1}\right)^2 . \tag{4.6}$$

2. Convolution of Exponentials

The following result, for the convolution of independent Exponential densities, will be used in deriving the distribution of S_j. See, e.g., Feller [Ref. 8].

Let $X_1, \ldots, X_j$ have densities

$$f_{X_i}(t) = \lambda_i e^{-\lambda_i t}\ ,\ t>0\ ;\quad \lambda_i \neq \lambda_k\ ,\ \forall i \neq k . \tag{4.7}$$

Then the sum $X_1 + \ldots + X_j$ has density:

$$f_{S_j}(t) = \left(\prod_{i=1}^{j}\lambda_i\right)\left(\sum_{i=1}^{j} B_{i,j}\, e^{-\lambda_i t}\right); \tag{4.8}$$

with the $B_{i,j}$ defined by the following product:

$$B_{i,j} = \prod_{\substack{k=1 \\ k\neq i}}^{j}(\lambda_k - \lambda_i)^{-1} . \tag{4.9}$$

3. Derivation of the PMF of N(T)

In the current case S_j, as defined in (4.5), is a linear combination of j *i.i.d. exponential* distributions, as seen in section III.B.1. Also by a simple transformation of scale:

$$\frac{j-i+1}{n-i+1} Y_i \sim \textit{Exponential}\left(\frac{n-i+1}{j-i+1}\lambda\right), \tag{4.10}$$

thus S_j can be viewed as a sum of j independent *exponential* random variables with different rates. Then applying result (4.8) yields the distribution, after some algebra:

$$f_{S_j}(t) = \lambda\binom{n}{j}\sum_{i=1}^{j}\prod_{\substack{k=1\\k\neq i}}^{j}\left(\frac{n-k+1}{j-k+1}-\frac{n-i+1}{j-i+1}\right)^{-1} e^{-\frac{n-i+1}{j-i+1}\lambda t}\ ; \tag{4.11}$$

$$j = 1,\ldots,n-1\ ;\ \lambda > 0\ ;\ t > 0\ .$$

To derive the PMF of N(T) the density above is used:

$$\begin{aligned} P(N(T)=j) &= \bar{F}_{S_{j-1}}(T) - \bar{F}_{S_j}(T)\ ; \\ \textit{with } \bar{F}_{S_j}(T) &= P(S_j > T) = 1 - F_{S_j}(T)\ , \end{aligned} \tag{4.12}$$

where $F_{S_j}(t)$ is the CDF of S_j and $\bar{F}_{S_j}(t)$ is obtained from the density (4.11) by integration:

$$\begin{aligned} \bar{F}_{S_j}(T) &= \int_T^{\infty} f_{S_j}(\tau)d\tau \\ &= \binom{n}{j}\sum_{i=1}^{j}\left[\prod_{\substack{k=1\\k\neq i}}^{j}\left(\frac{n-k+j}{j-k+1}-\frac{n-i+1}{j-i+1}\right)^{-1}\right]\frac{j-i+1}{n-i+1}e^{-\frac{n-i+1}{j-i+1}\lambda T}\ ; \end{aligned} \tag{4.13}$$

$$\textit{defined for } j = 1,\ \ldots\ ,\ n-1\ .$$

Using (4.12) we have the desired PMF for $j = 1,\ldots,n-2$. The values $j = 0, n-1, n$ have not been covered so far.

$$P(N(T)=n) = P(\sum_{i=1}^{n} X_i \le T) = F_{S_n}(T) ; \tag{4.14}$$

since S_n is a sum of *i.i.d. exponential* Random Variables, then:

$$\begin{aligned} & S_n \sim Gamma(n, \lambda) , \lambda > 0 ; \\ \Rightarrow & P(N(T)=n) = 1 - e^{-\lambda T} \sum_{j=0}^{n-1} \frac{(\lambda T)^j}{j!} ; \end{aligned} \tag{4.15}$$

If no task is completed, the shortest time to complete a task must be bigger that the time window, i.e.:

$$P(N(T)=0) = \bar{F}_{S_1}(T) = e^{-n\lambda T} . \tag{4.16}$$

Then by a generalization of the domain of definition in (4.13),

$$\bar{F}_{S_j}(T) = \begin{cases} 0 & if\ j \le 0 \\ (4.13) & if\ j = 1, \ldots, n-1 \\ e^{-\lambda T} \sum_{j=0}^{n-1} \frac{(\lambda T)^j}{j!} & if\ j = n \\ 1 & if\ j \ge n+1 \end{cases} . \tag{4.17}$$

With this broader definition of $\bar{F}_{S_j}(T)$ (4.12) holds for the whole range of $j, \{0, 1, \ldots, n\}$, and an exact PMF of $N(T)$ has been derived.

C. APPROXIMATION TO THE PMF OF N(T)

To recapitulate, an exact but complicated formula for the PMF of $N(T)$ has been derived earlier; see (4.17). We now desire an approximation, which might be

easier to use. This can be achieved via the distribution (CDF) of S_j in the following way:

$$\begin{aligned} P(N(T) = j) &= P(N(T) < j+1) - P(N(T) < j) \\ &= F_{S_j}(T) - F_{S_{j-1}}(T) . \end{aligned} \tag{4.18}$$

1. Gamma Approximation

Analysis of simulation data, e.g. using probability plots, suggest that the S_j are approximately distributed *Gamma* (see Appendix C for details). If the approximation is good enough it can be used to compute values of $P(N(T) = j)$. The values of the Expectation and Variance of the S_j are available in (4.6).

If S_j is to be distributed *Gamma* with parameters shape α_j and rate β_j, then:

$$E[S_j] = \frac{\alpha_j}{\beta_j} \quad \textit{and} \quad Var[S_j] = \frac{\alpha_j}{\beta_j^2} ; \tag{4.19}$$

from which the parameter values of the approximating distribution can be derived, since the values of the expectation and variance are known:

$$\alpha_j = \left(\sum_{i=1}^{j} \frac{j-i+1}{n-i+1}\right)^2 / \sum_{i=1}^{j}\left(\frac{j-i+1}{n-i+1}\right)^2 ,$$
$$\beta_j = \lambda \sum_{i=1}^{j} \frac{j-i+1}{n-i+1} / \sum_{i=1}^{j}\left(\frac{j-i+1}{n-i+1}\right)^2 . \qquad (4.20)$$

It was seen that this approximation produces good results in computing the PMF of $N(T)$ using (4.18). Another approximation method is presented next.

2. Normal Approximation

If we assume that S_j can be represented by a *Normal* distribution then the resulting Normal approximation,with $\Phi(\,)$ denoting the probability mass in the left tail of a Standard Normal, is:

$$P(N(T)=j) = \Phi\left(\frac{T-E[S_j]}{\sqrt{Var[S_j]}}\right) - \Phi\left(\frac{T-E[S_{j+1}]}{\sqrt{Var[S_{j+1}]}}\right), \qquad (4.21)$$

for $j = 1, \ldots, n-1$; and with $E[S_j]$, $Var[S_j]$ defined by (4.6).

D. RESULTS

Once more analytical results were checked through a Monte Carlo simulation. The simulation program replicated independent "packing experiments". The values of "Number Packed" (or tasks completed) thus obtained, were then used in an APL function to compute the "experimental" distribution of $N(T)$. This distribution is plotted together with the corresponding approximations and exact PMF, (4.12). The inclusion of a continuity correction in the Normal Approximation provided no substantial improvement. The Normal approximation was seen to work better than portrayed in Figure 4.1, for cases with bigger values of n, as should be expected.

The Gamma approximation produced results quite similar to those of the exact formula.

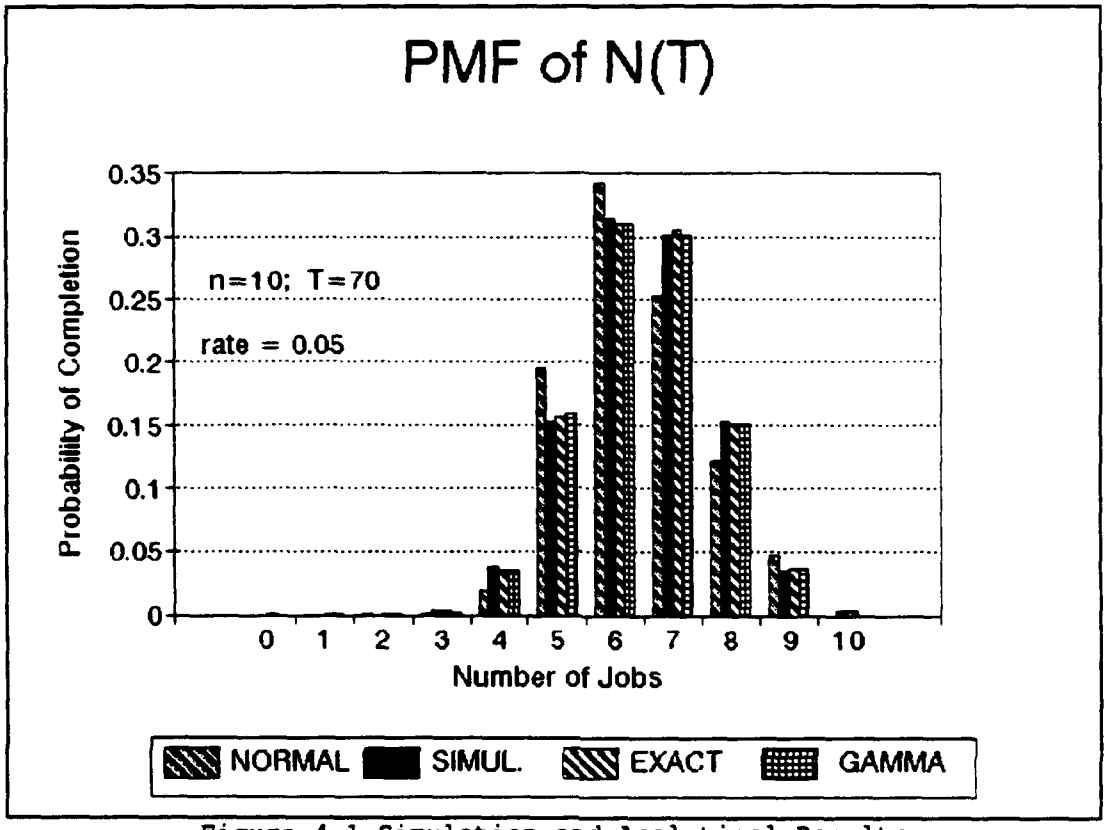

Figure 4.1 Simulation and Analytical Results

Suppose an unsophisticated policy is used, so jobs are taken randomly, without ordering. In that case $N(T) \sim Poisson(\lambda)$, and $E[N(T)] = \lambda T$. The PMF of the number of jobs completed in this way, was compared with the corresponding PMF under a SF rule. This gives a measure of the benefits of employing the latter policy. Results are summarized in Table 4.1. It can be seen in the bottom row that, for this particular set of parameters, the SF rule accounts for an increase in Expected value of about 85%.

Table 4.1 COMPARISON OF RESULTS
FOR THE DISTRIBUTION OF THE NUMBER OF TASKS COMPLETED

j	SIMULAT	NORMAL	GAMMA	EXACT	POISSON
0	0	-	-	0	0.030197
1	0	0	0	0	0.105691
2	0	0	0.000031	0.000056	0.184959
3	0.002800	0.000034	0.002531	0.002934	0.215785
4	0.037800	0.019231	0.035266	0.035456	0.188812
5	0.152300	0.195998	0.158881	0.156441	0.132169
6	0.315000	0.342860	0.310115	0.309978	0.077098
7	0.301300	0.252402	0.301318	0.303980	0.038549
8	0.152500	0.122254	0.151571	0.151470	0.016866
9	0.035700	0.047305	0.036936	0.036370	0.006559
10	0.002600	-	-	0.003315	0.002296
AVG	6.487500	6.284765	6.458033	6.493038	3.500000

E. APPLICATION TO MISSILE DEFENSE

We now proceed to incorporate the results obtained from the packing problem to a missile attack situation. As before, the defender must acquire and shoot the maximum number possible of n incoming missiles, within a restricted time window.

1. Assumptions

The missiles are acquired with probability p, acquisitions being independent *Bernoulli* trials. A fixed time, w, is spent for each acquisition. The defender, knowing the ordering of the "missile killing times", will select them for acquisition with a SF rule. It is not recognized if acquisitions are successful or not. Consequently, the first time the defender fails to acquire a missile the residual time window is expended and no more missiles are shot. Presumably the performance

could be improved by introducing a threshold, as was done earlier, but this option has not been studied.

2. Model

Let $N(T)$ now represent the number of missiles killed in time T, under the above assumptions; then,

$$P(N(T) \geq j) = p^j F_{S_j}(T - jw); \tag{4.22}$$

resulting in the PMF:

$$P(N(T) = j) = p^j F_{S_j}(T - jw) - p^{j+1} F_{S_{j+1}}(T - (j+1)w). \tag{4.23}$$

Recalling that $F_{S_j}(t) = 1 - \bar{F}_{S_j}(t)$ can be obtained from (4.13).

The result (4.23) applied to a given set of parameters is plotted in Figure 4.2 together with corresponding values obtained through simulation.

For the purpose of comparison the results of the model were checked against those obtained from employing the unsophisticated strategy, i.e., taking the missiles in random order. In this case the distribution of $N(T)$ is obtained in the same fashion, by using (4.23), but with the difference that now S_j is distributed *Gamma* with shape j and rate λ. The model is seen to agree reasonably well with the "experimental simulation data" as further detailed in Table 4.2. It is also verified a substantial improvement by using the SF strategy.

A noticeable feature of the distribution is that it is no longer unimodal, unlike similar distributions for the packing problem. In plots generated with different rate values one local minimum appeared between two maxima, one of them located at zero. Figure 4.3 depicts a situation in which the rate is 0.1 and 20 missiles are present, all other parameters remaining equal to the previous situation. As can be

seen the minimum is now more sharply defined, and the absolute maximum is at zero.

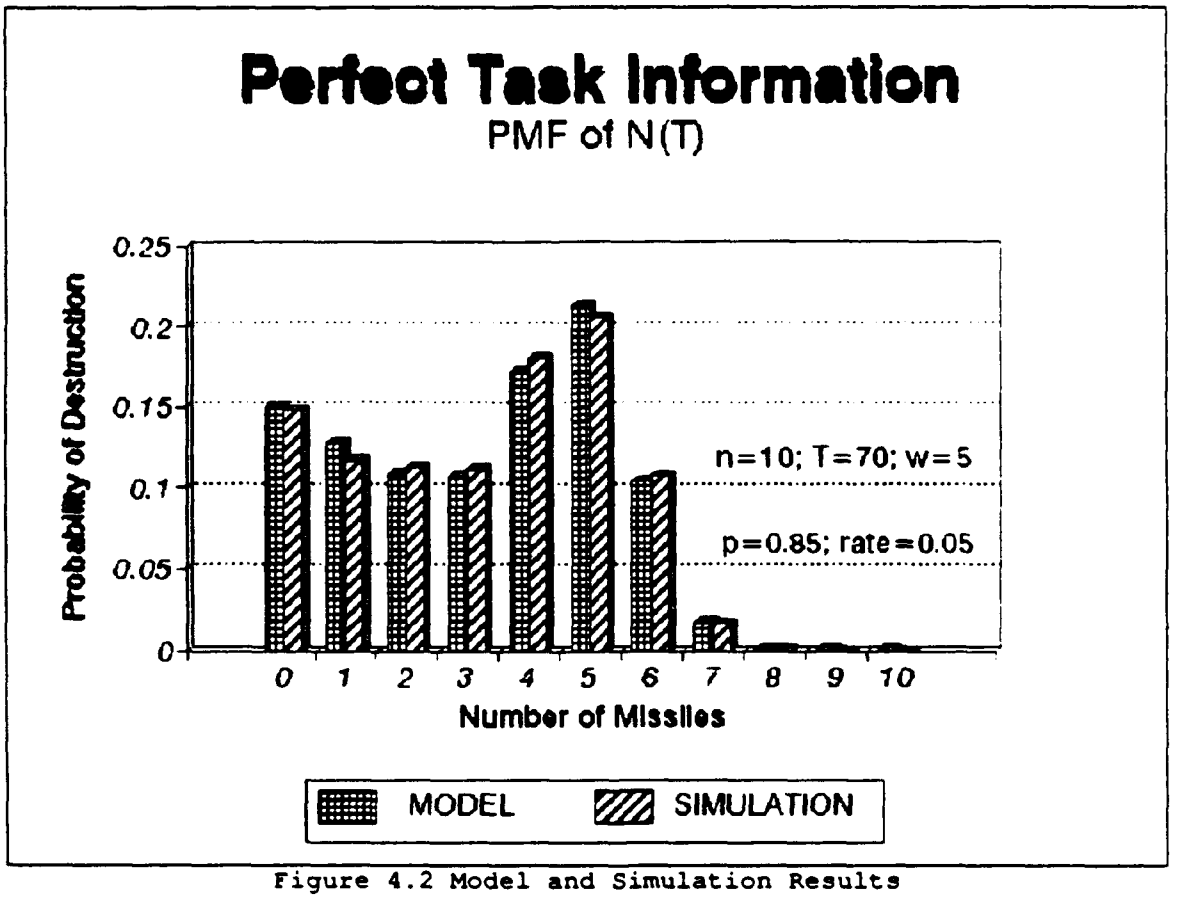

Figure 4.2 Model and Simulation Results

Table 4.2 PERFECT TASK INFORMATION PMF

j	SIMULATION	MODEL	NO STRATEGY
0	0.148574	0.150000	0.182953
1	0.117059	0.127500	0.238861
2	0.113057	0.108782	0.259735
3	0.111056	0.106964	0.191905
4	0.180590	0.171499	0.091931
5	0.205103	0.212791	0.028368
6	0.107554	0.104298	0.005541
7	0.016508	0.017283	0.000659
8	0.000500	0.000872	0.000045
9	0	0.000011	0.000002
10	0	0	0
AVG	3.189096	3.169751	1.881847

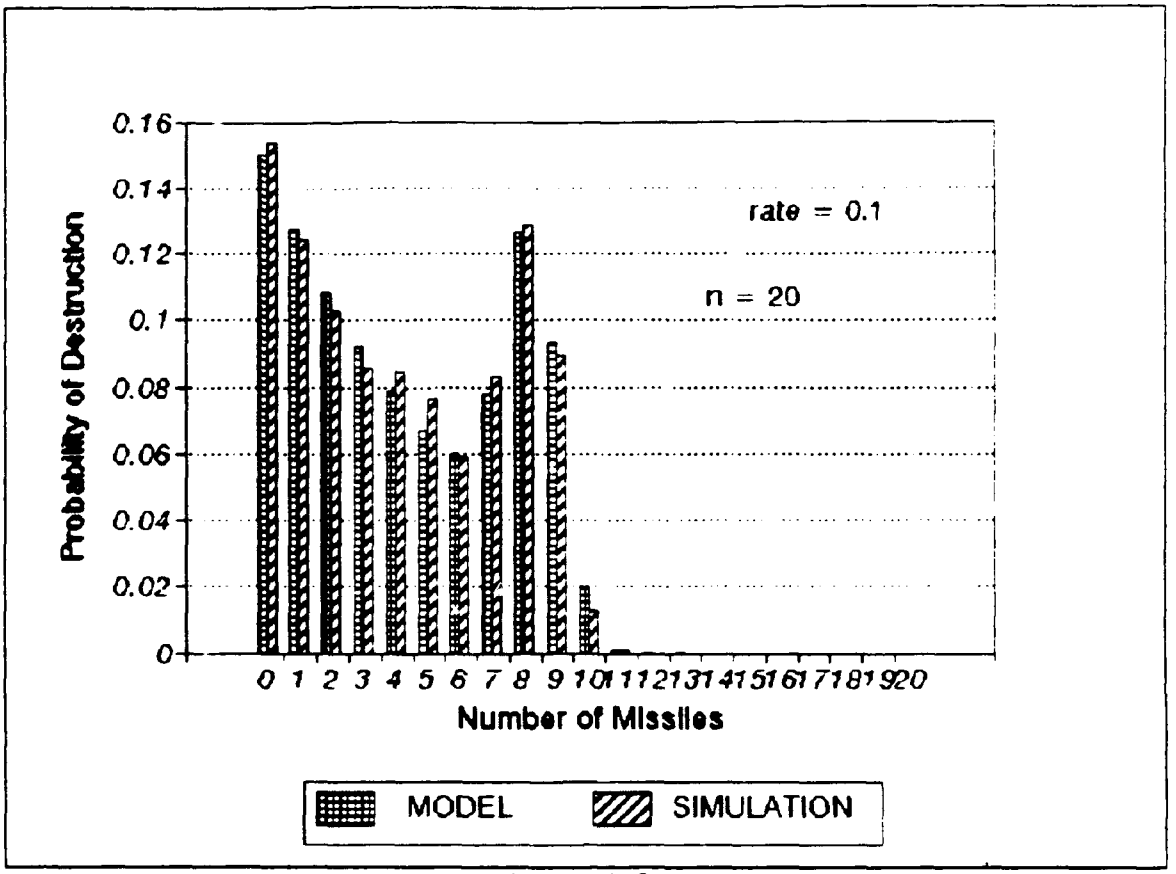
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
Probability of Destruction
rate = 0.1
n = 20
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Number of Missiles
MODEL
SIMULATION

Figure 4.3

V. CONCLUSIONS

A. MODEL VERIFICATION

In the preceding chapters we developed three models for defense against missile attacks. For each model analytic results were compared with simulations, for some range or combination of parameters. To summarize, our three models correspond to the following situations, in ascending order of information availability:

Invisible Kill

Unknown: If acquisition successful.
If and when missiles shot.
Strategy: Allocate time evenly between all engaged missiles.
Decision: How many missiles to engage.

Visible Kill

Unknown: If acquisition successful.
Known: Time when missile killed (if killed).
Strategy: Allocate a max threshold time to each missile engagement.
Decision: Length of threshold.

Perfect Task Information

Unknown: If acquisition successful.
Known: Times of destruction of each missile.
Strategy: Engage missiles with SF rule.

Throughout our work, whenever numerical results were necessary, assumptions on the values of the parameters had to be taken. We have used what seemed to be "reasonable" values of killing rates, acquisition times, time window, etc. These, however, may differ from actual parameter values to be found in existing weapon systems. For this reason we now seek to "validate" the models in more general conditions. We wish to verify and compare our models in a reasonable number of parameter (factors) combination without too much simulation effort. The response

variable to consider for validation/comparison purposes will be the expected number of missiles destroyed. The models depend on five input parameters or factors: n, p, w, λ, T. The same distribution for the Killing Time will be used - an *Exponential*. In order to generate input parameter values we chose a two level fractional factorial design, 2_V^{5-1}, [Ref. 9], which seems to attain the above stated objectives. This design corresponds to the input data set displayed on Table 5.1.

Table 5.1 INPUT DATA

RUN	λ	n	p	w	T
1	0.05	50	1.0	15.0	70.0
2	0.1	10	1.0	5.0	200.0
3	0.1	50	1.0	15.0	200.0
4	0.05	10	0.85	5.0	200.0
5	0.1	10	0.85	5.0	70.0
6	0.1	50	0.85	15.0	70.0
7	0.05	10	1.0	5.0	70.0
8	0.1	10	1.0	15.0	70.0
9	0.05	10	0.85	15.0	70.0
10	0.1	10	0.85	15.0	200.0
11	0.05	10	1.0	15.0	200.0
12	0.05	50	0.85	5.0	70.0
13	0.05	50	1.0	5.0	200.0
14	0.1	50	0.85	5.0	200.0
15	0.1	50	1.0	5.0	70.0
16	0.05	50	0.85	15.0	200.0

1. Invisible Kill

The simulation was carried out in the following way: For each run, the optimal value of the number of missiles to engage was computed according to the model equation (2.11). In each run the simulation replicates enough independent attack experiments to get estimates of the average number of missiles killed and 95% confidence bounds. The same average was computed from the model equation (2.13). Results are compared in Table 5.2.

Table 5.2 INVISIBLE KILL VERIFICATION

RUN	LOWER	AVERAGE	UPPER	MODEL
1	1.248880	1.260090	1.271298	1.264240
2	7.711665	7.736362	7.761059	7.768698
3	2.981409	3.005450	3.029490	2.997817
4	4.461138	4.487008	4.512877	4.484883
5	2.301197	2.321421	2.341643	2.341143
6	0.746798	0.759386	0.771993	0.752077
7	1.819565	1.835803	1.852040	1.858953
8	0.867102	0.880358	0.893614	0.884797
9	1.060356	1.071748	1.083138	1.074604
10	3.318501	3.342890	3.367278	3.344489
11	3.099832	3.122495	3.145157	3.147755
12	1.544501	1.560270	1.576038	1.580111
13	5.153650	5.183533	5.213414	5.199824
14	6.658032	6.693637	6.729240	6.688979
15	2.723549	2.744591	2.765632	2.757285
16	2.660858	2.682709	2.704558	2.675591

As further depicted in Figure 5.1, all runs produced satisfying results. Considering that (2.13) is an approximation, the model values seem to be in accordance with those of the simulation.

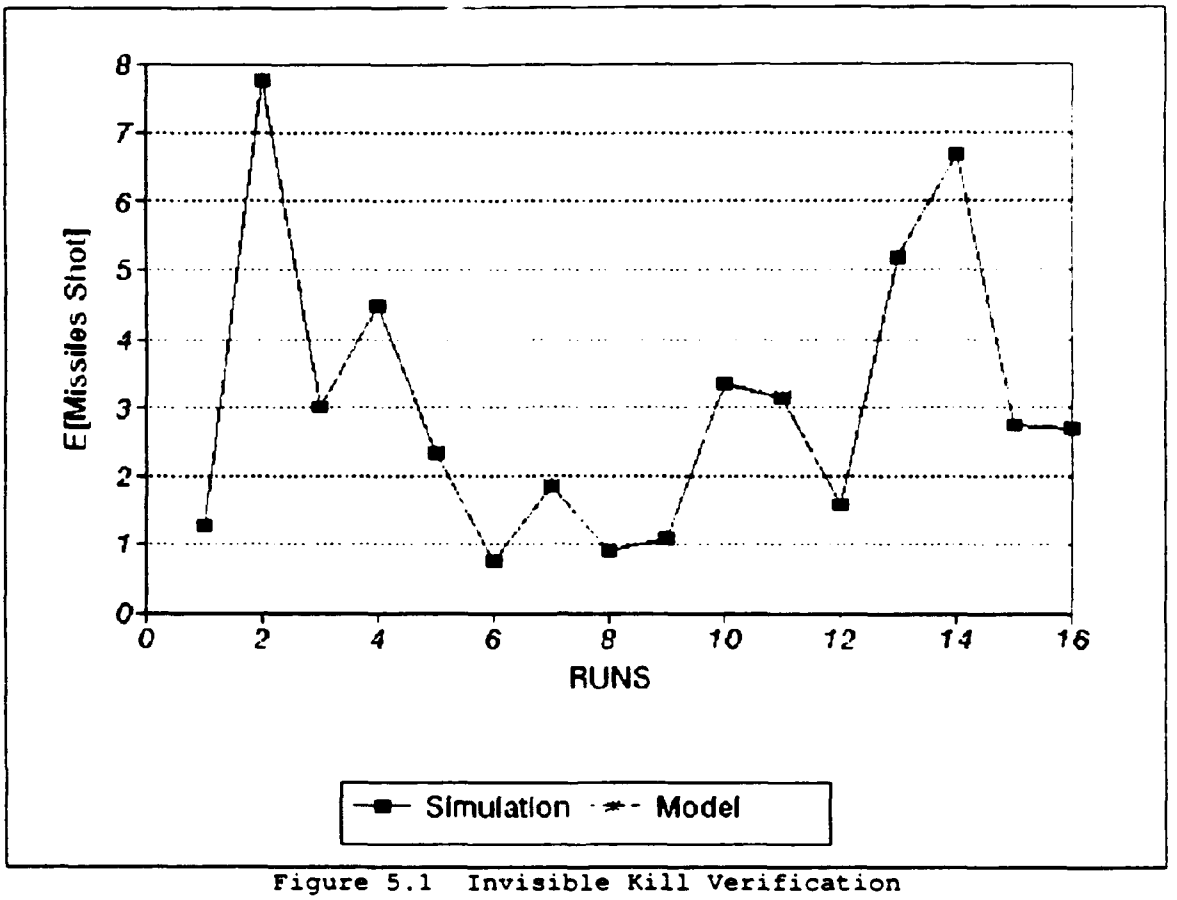

Figure 5.1 Invisible Kill Verification

2. Visible Kill - Threshold Strategy

We first determined the optimal threshold for each run by numerical solution of (3.10), (when $p = 1, q = 0$ the optimal threshold will be T). The simulation is executed for each combination of factors (run) and optimal threshold. Analytical values (model) are obtained from (3.8). As can be seen in the following table and figure, the results obtained from the model are not always accurate as compared to the simulation. In fact the model predicts an expectation of about 13

missiles destroyed, when only 10 are present in the attacks corresponding to run 2. This is an expected result considering the crude approximations taken in Chapter III. Otherwise we find the agreement model/simulation to be quite acceptable.

Table 5.3 VISIBLE KILL VERIFICATION

RUN	LOWER	AVERAGE	UPPER	MODEL
1	2.132568	2.148101	2.163632	1.973662
2	9.907267	9.924894	9.942521	13.333333
3	7.941874	7.966734	7.991593	7.999999
4	5.966601	6.009499	6.052395	5.819679
5	3.540552	3.572495	3.604436	3.356913
6	1.996812	2.016756	2.036670	2.057663
7	3.296708	3.323905	3.351101	2.782671
8	2.754568	2.767375	2.780181	2.798467
9	1.479404	1.500816	1.522227	1.441650
10	5.823129	5.858012	5.892895	5.879036
11	5.904632	5.930957	5.957281	5.714175
12	2.212835	2.241029	2.269221	2.036888
13	8.536298	8.581685	8.627071	7.999927
14	9.724030	9.779253	9.834475	9.591181
15	5.016041	5.043329	5.070617	4.665247
16	4.205375	4.238362	4.271349	4.119000

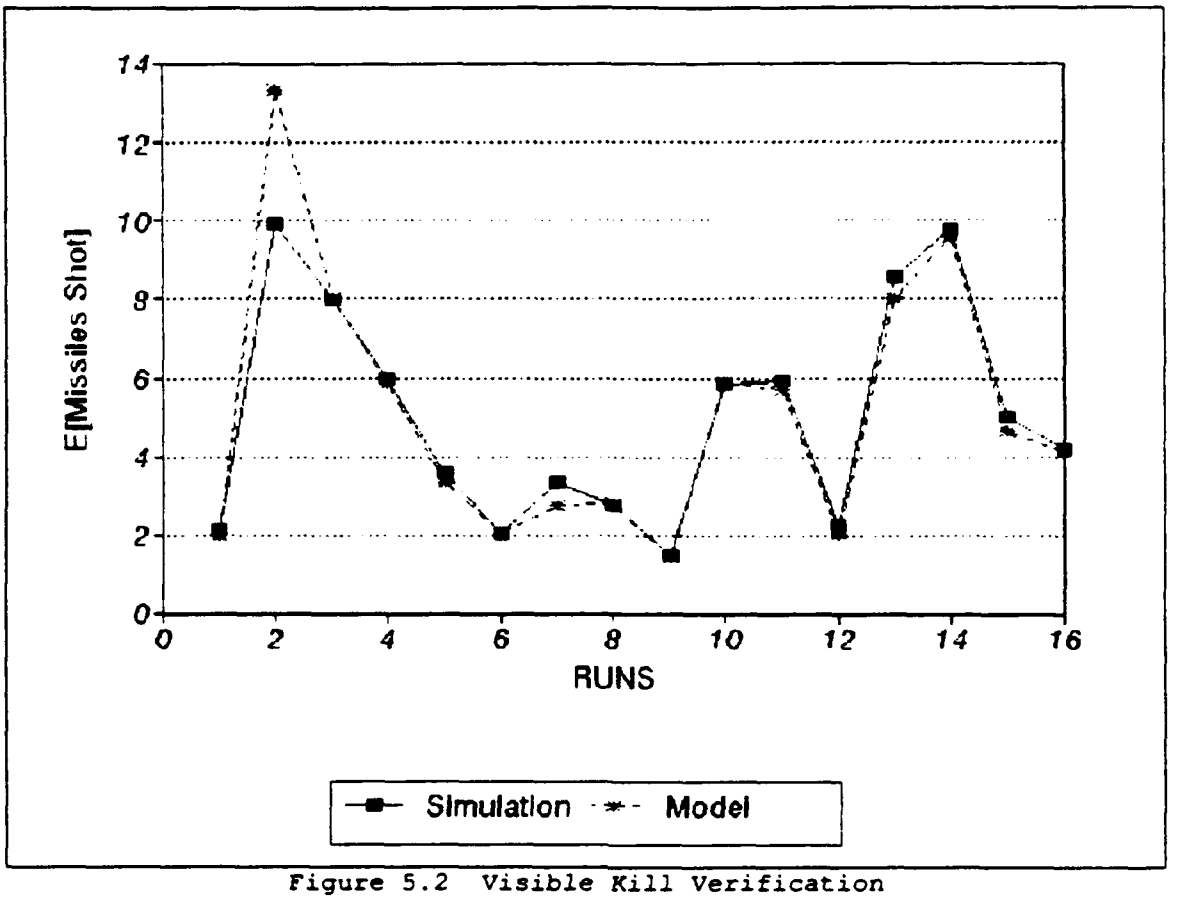

Figure 5.2 Visible Kill Verification

3. Perfect Task Information

Proceeding in an analogous way we compare the expected number of missiles shot obtained from simulation, with model results computed with:

$$E[N(T)] = \sum_{j=0}^{n} jP(N(T) = j); \tag{5.1}$$

with $P(N(T) = j)$ being given in (4.23).

The fit between model and simulation is seen to be excellent in this case.

Table 5.4 PERFECT TASK INFORMATION VERIFICATION

RUN	LOWER	AVERAGE	UPPER	MODEL
1	3.972124	3.978036	3.983947	3.981028
2	9.888283	9.905581	9.922877	9.927288
3	11.701859	11.723443	11.745027	11.743412
4	4.248149	4.312770	4.377340	4.294308
5	3.632813	3.682852	3.732889	3.672681
6	2.672961	2.703552	2.734142	2.708608
7	5.097260	5.116289	5.135318	5.110362
8	3.470990	3.481924	3.492857	3.484863
9	2.161409	2.185453	2.209497	2.198917
10	4.157371	4.219392	4.281412	4.211194
11	7.053313	7.071574	7.089834	7.088154
12	4.354237	4.421083	4.487927	4.388366
13	19.723969	19.771378	19.818771	20.385315
14	5.409484	5.520819	5.632153	5.554571
15	10.613348	10.635718	10.658088	10.652422
16	4.589483	4.664865	4.740247	4.691323

B. COMPARISON OF STRATEGIES

Given that not all the models produced equally reliable results, we chose to compare the different strategies using simulation. Again, the input data set of Table 5.1 is used to generate the values depicted in Figure 5.4.

Clearly the SF strategy in the Perfect Task Information situation shows the best overall performance. This is not a surprising result, since this strategy benefits from the highest level of information. Also the Threshold strategy comes next in overall performance, and Invisible Kill produces the worst results. More unexpected is the

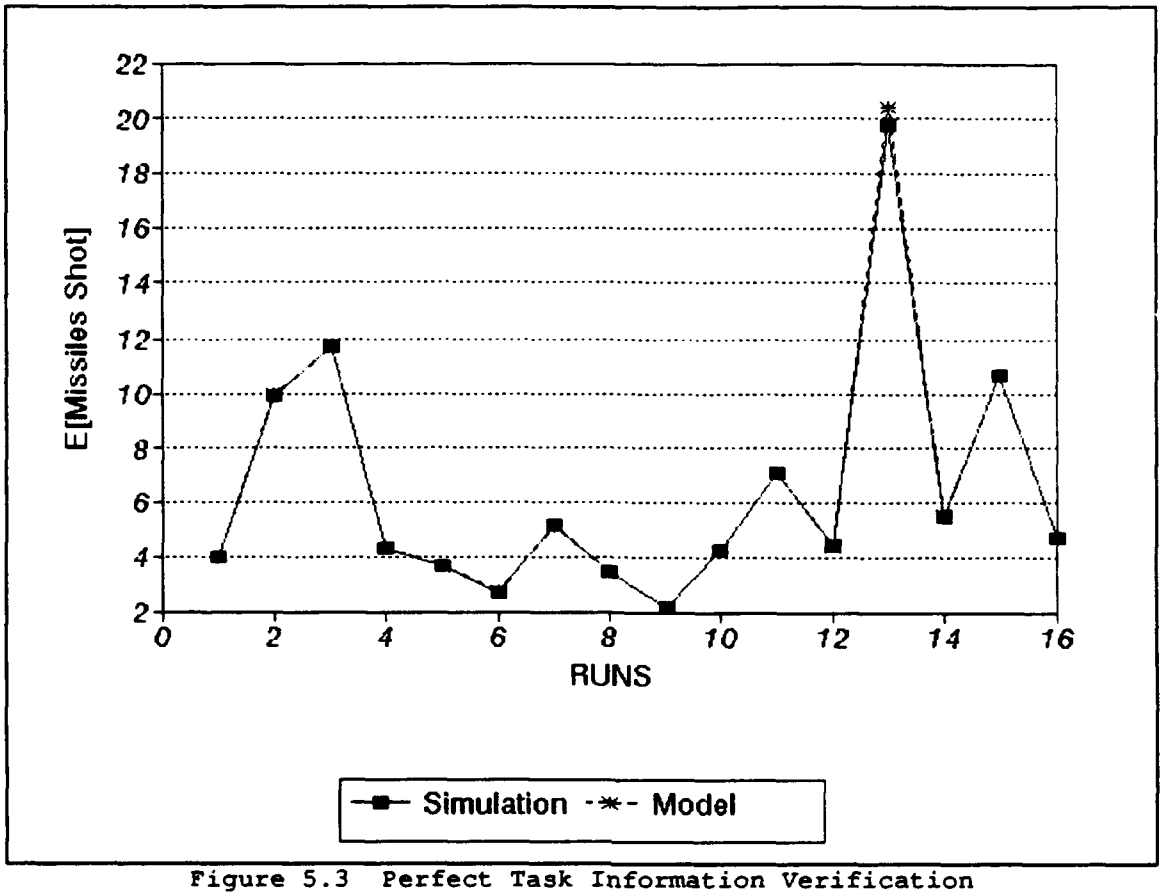

Figure 5.3 Perfect Task Information Verification

fact that in three cases (runs 4, 10, 14) Threshold wins from Perfect Task Information, which is even outperformed by the modest Invisible Kill in run 14. Although these exceptions do not alter the conclusion that models with access to more information perform better overall, it is somewhat surprising that this situation does not occur for every combination of parameters. A closer look at the conspicuous runs reveals that in all of them $p = 0.85$, $T = 200$. Thus the explanation for the underperformance of Perfect Task Information must lie in the fact that acquisition is still invisible (see IV.E.1). This implies that, given sufficient time, the defender eventually fails to acquire. Unlike with Threshold and Invisible Kill there is no chance of reacquisition. However, we see that in more favorable conditions, particularly when acquisition occurs with probability one, the SF rule can enhance performance up to five times (run 13).

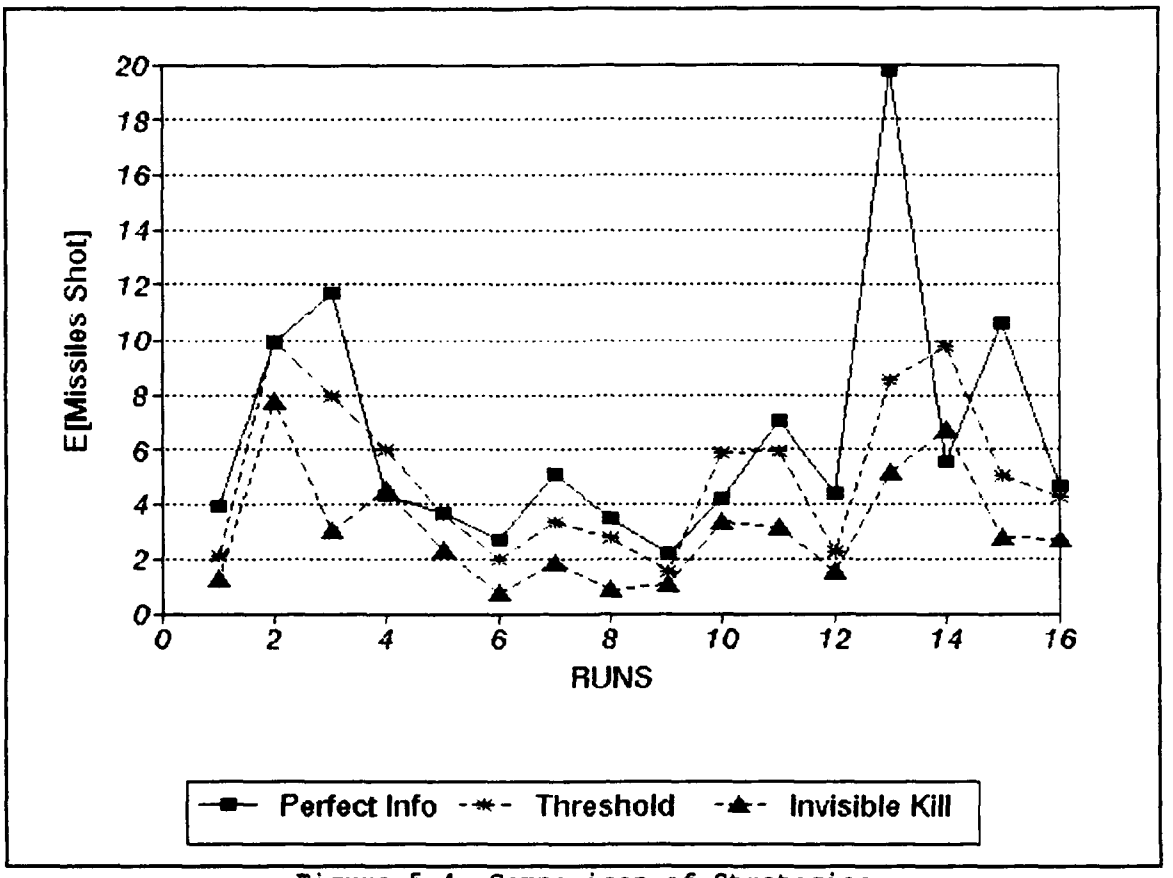

Figure 5.4 Comparison of Strategies

The shortcomings in the Perfect Task Information strategy, can be overcome by refining the model assumptions in IV.E.1. The proposed refinement consists on including a "threshold" time, such that, after this time has elapsed without target destruction being observed, the defender reattempts acquisition of the same target. Various thresholds can be used (e.g., the largest task time). The most natural thresholds will be the known task times (in this case, knowledge only of the ordering is not sufficient).

With this improvement the Perfect Task Information model clearly dominates for all simulated runs, as Figure 5.5 shows. We do not provide analytical results for this improved model.

Translated into a context where the weapons of the defender consist on interceptor missiles, these findings seem to strongly favor a system that enables the defender to assign each interceptor according to the shortest time for interception (SF).

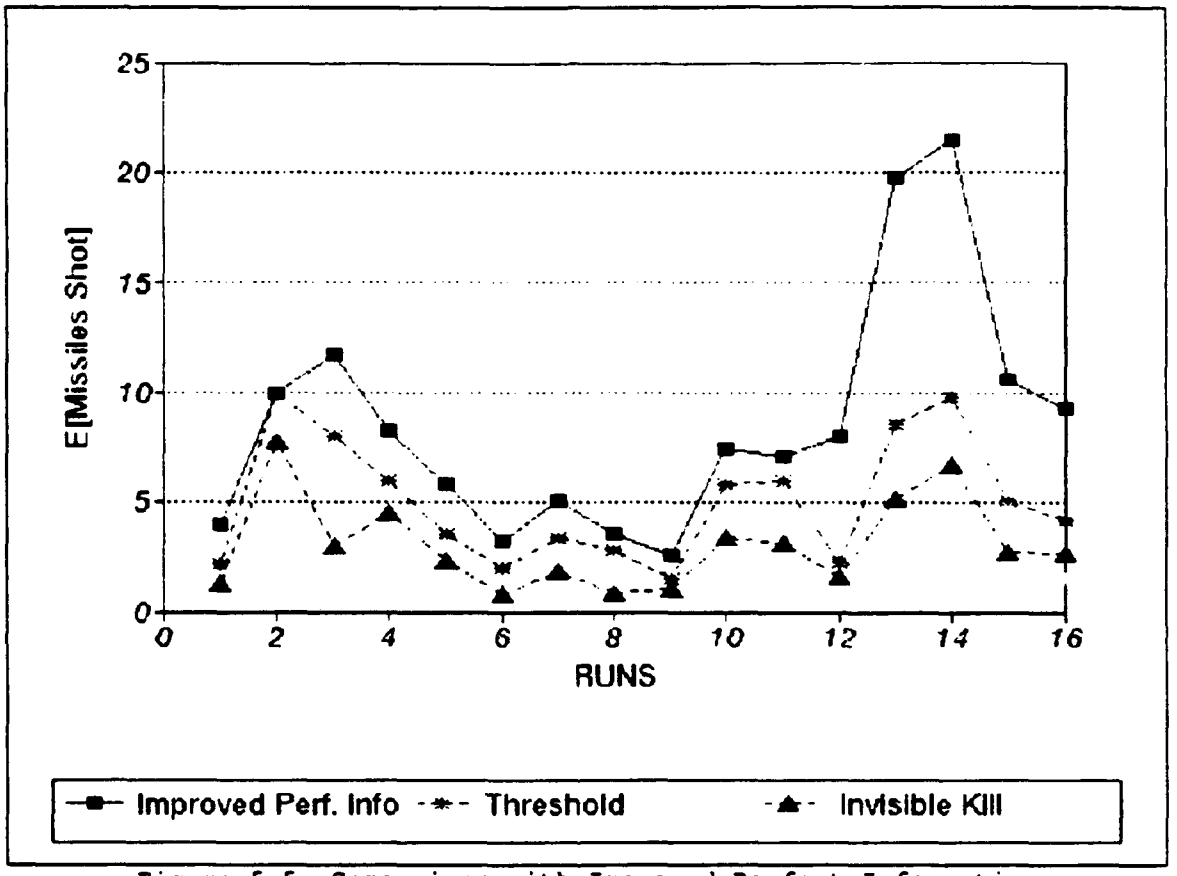

Figure 5.5 Comparison with Improved Perfect Information

C. APPLICATION EXAMPLE

Suppose one is planning a weapon system to face a threat that allows a time window T, and that we can forecast that n missiles will be used in each attack. We have chosen one of the three models to be the most adequate to describe the weapon system that can be built. Then we have, from our analytic work, the function:

$$E[N(T)] = f(\lambda, p, w; T, n). \tag{5.2}$$

The arguments T and n are dictated by the threat, while λ, p, w can be incorporated in the design of the defense system, within technical feasibility and resource constraints, and are thus subject to decision. Further, suppose we can estimate the cost functions of increasing the probability of acquisition , c_p, increasing the rate of kill , c_λ, and decreasing the acquisition time , c_w. The decision for the optimal combination of parameter to design into the system can be obtained by solving the nonlinear program:

$$\begin{aligned} & Max \quad E[N(T)] \\ s.t. \quad & c_\lambda(\lambda) + c_p(p) + c_w(w) \le R \\ & 0 < \lambda \le \lambda_{max} \\ & 0 < p < 1 \\ & 0 < w \le w_{max}. \end{aligned} \tag{5.3}$$

D. RECOMMENDATIONS

The models developed in this thesis may provide a basis for further work leading to realistic models of missile attack situations. For this objective to be attained, "real world" data and system specific information must be used to validate the models. Further investigation may be needed to verify the nature of the distribution of the killing times, or the robustness of the models to distributional assumptions.

A possible refinement would be to drop the assumption of invisible acquisition which is common to all three models but heavily penalizes the model of Perfect Task Information.

It might also be useful to consider other objective functions. For example, one may argue that in a scenario where some missiles are almost certain to leak, it makes more sense to attempt to minimize the maximum number of leaking missiles. Under some circumstances it may be desirable to maximize the probability of killing all attacking missiles; under others it may be useful overall to expend some resources to pass information about leaking missiles (course, estimated time of arrival) to a secondary defense level, D2. There are numerous other options and opportunities.

APPENDIX A. FORTRAN CODES

In this appendix we list the most important programs and routines. The codes are written in Fortran77 and were run on an Amdahl 5990-5000, at the NPS Computer Center.

The simulation programs shown were used to verify the numerical results obtained with the probability models. The version of the Perfect Task Information model presented is the one without the threshold improvement. A program to generate exact (analytical) results of the Perfect Task Information is also included. Exact computations for the other two models are simple and can be done on a programmable calculator.

All simulations terminate either with the destruction of the last missile or when the window elapses.

A. INVISIBLE KILL

The main program, "Mass Attack with Invisible Kill", implements the simulation to verify the model developed in Chapter II. The program executes independent replications of the simulation of a missile attack, to compute the expected number of missiles killed. This is done for the 16 sets of parameters displayed in Table 5.1, and with m^o found as shown in Chapter II.

Two event routines are present: "Acquisition" and "Shooting". The simulation also requires the SIMUTIL package [Ref. 2], LLRANDOMII [Ref. 3], and subroutine STAT.

A descriptive list of the variables follows:

- "ProbAcq" is the probability of acquisition denoted by p in Chapter II.
- "TimeWindow" is the time available to destroy the targets, T.
- "TimeAcq" is the fixed acquisition or setup time, w.
- "TimetoKill" is the time allowed to shoot down a single target.
- "Rate" is the killing rate, and also represents the rate parameter of an exponential distribution, λ.
- "Average", "Variance" and "IntervalWidth" are statistics of the number of missiles killed per attack, taken from the sample of all the simulated attacks. These statistics are computed by subroutine STAT.
- "Lower" and "Upper" are the 95% confidence bounds for the expectation of the number of missiles killed.
- "Iseed" is the seed for the random number generator. It is initialized through a "Read" statement at the beginning of the simulation.
- "NrMissiles" is the number of missiles in an attack, n.
- "m" is the optimal number of missiles to engage. It is computed through approximation (2.11), as described in Chapter II. It is an input parameter to the simulation.
- "Processed" counts missiles as they are generated.
- "NrAcquired" counts the number of successful acquisitions.
- "MissilesDown" counts the number of missiles shot. It is used as an input by subroutine "STAT" to compute the average number of missiles killed.
- The integer variables "i" and "k" are indices used in "Do" loops.
- "Acquired" is a logical variable and has value .True. when the missile being processed has been successfully acquired.

- "Destroyed" is a logical variable. It has value .True. when the missile being processed is destroyed.

- "Tsample" is the random time to destroy a missile sampled from an exponential distribution with rate λ.

- "Now" is the current simulation time. It is controlled by subroutine SIMGO [Ref. 2].

The Fortran77 code for the simulation is as follows:

```
      PROGRAM Mass Attack with Invisible Kill
*
      REAL ProbAcq, TimeWindow, TimeAcq, TimetoKill, Rate, Average,
     +Variance, IntervalWidth, Lower, Upper
      INTEGER Iseed, NrMissiles, m, Processed, NrAcquired, k,
     +MissilesDown, i
*
      COMMON/ SHARE2/ Processed, NrAcquired, MissilesDown
      COMMON/ SHARE3/ Iseed, ProbAcq, TimeAcq, TimetoKill, Rate,
     +NrMissiles
      COMMON/ SHARE4/ TimeWindow
      COMMON/ SHARE5/ Average, Variance, IntervalWidth, Lower, Upper
*
*     Create input and output files.
      OPEN( 15, FILE='/EXPINV DATA')
      OPEN( 20, FILE='/EXPINV OUTPUT')
*
*     Initialize the seed for the random number generator
      READ(15, *) Iseed
      Do i = 1, 16
*     Read one combination of input parameters (see Table 5.1), and replicate the
*     simulation for each of them.
        Read (15,*) Rate, NrMissiles, ProbAcq, TimeAcq, TimeWindow, m
*       The time assigned to shoot at each missile is computed from the optimal
*       number of missiles killed according to equation (2.2).
        If( m .ne. 0 ) then
          TimetoKill = TimeWindow / m - TimeAcq
        Else
          TimetoKill = 0.0
```

```
      End If
*
*     Initialize the statistics.
      Variance = 0.0
      Average = 0.0
      IntervalWidth = 1.0
*
*     Perform independent replications of the simulation.
      Do k = 1, 15000
        CALL INIT
        CALL SIMGO
        CALL STAT( MissilesDown, k)
      End Do
      Write(20,*) i, Lower, Average, Upper
    End Do
*
    CLOSE(15)
    CLOSE(20)
*
    Stop
*
    End

    SUBROUTINE INIT
*
*   Initialize variables used in the simulation before each replication.
    REAL Now, TimeWindow
    INTEGER Processed, NrAcquired, MissilesDown
*
    COMMON/ SHARE1/ Now
    COMMON/ SHARE2/ Processed, NrAcquired, MissilesDown
    COMMON/ SHARE4/ TimeWindow
*
    Now = 0.0
    Processed = 0
    NrAcquired = 0
    MissilesDown = 0
*
    CALL CALANDAR INITIALIZATION
*
*   Schedule the first acquisition event
```

```
      CALL SCHEDULE( Now, 1)
*
*     Schedule end of attack after time window elapsed
      CALL SCHEDULE( Now+TimeWindow, 3)
*
      Return
*
      End
*
      SUBROUTINE ACQUISITION
*
      REAL X, ProbAcq, Now, TimeAcq, TimetoKill, Rate
      INTEGER Iseed, NrAcquired, Processed, MissilesDown, NrMissiles
      LOGICAL Acquired
*
      COMMON/ SHARE1/ Now
      COMMON/ SHARE2/ Processed, NrAcquired, MissilesDown
      COMMON/ SHARE3/ Iseed, ProbAcq, TimeAcq, TimetoKill, Rate,
     +NrMissiles
*
      SAVE SHARE2
*
*     Sample from a Uniform(0,1), to determine if acquisition was successful
      CALL RANNUM( 1, Iseed, 0.0, 1.0, 0.0, X)
      Acquired = .FALSE.
      If( X .le. ProbAcq ) Acquired = .TRUE.
      If( Acquired ) then
        NrAcquired = NrAcquired + 1
*     Schedule a shooting event for the acquired missile, after acquisition time
*     elapsed.
        CALL SCHEDULE( Now+TimeAcq, 2)
        Return
      else
        Processed = Processed + 1
        If( Processed .lt. NrMissiles ) then
*     If not acquired and not all missiles have been processed, schedule another
*     acquisition.
          CALL SCHEDULE( Now+TimeAcq+TimetoKill, 1)
        else
*     End the simulation if all missiles have been processed
          CALL SCHEDULE( Now+0.00000000001, 3)
```

```
        End If
      End If
*
      Return
*
      End
*
      SUBROUTINE SHOOTING
*
      REAL Now, TimeAcq, ProbAcq, Rate, Tsample, TimetoKill
      INTEGER MissilesDown, Processed, Iseed, NrAcquired, NrMissiles
      LOGICAL Destroyed
*
      COMMON/ SHARE1/ Now
      COMMON/ SHARE2/ Processed, NrAcquired, MissilesDown
      COMMON/ SHARE3/ Iseed, ProbAcq, TimeAcq, TimetoKill, Rate,
     +NrMissiles
*
      SAVE SHARE2
*
*     Samples the task time from an Exponential distribution
      CALL RANNUM( 3, Iseed, Rate, 0.0, 0.0, Tsample)
      Destroyed = .FALSE.

*     If the random task time is less than the time assigned to shoot the missile, it
*     will be destroyed.
      If( Tsample .le. TimetoKill ) Destroyed = .TRUE.
      If( Destroyed ) MissilesDown = MissilesDown + 1
      Processed = Processed + 1
      If( Processed .lt. NrMissiles ) then
*        Schedule another acquisition event
         CALL SCHEDULE( Now+TimetoKill, 1)
      else
*        if all missiles processed, end the attack
         CALL SCHEDULE( Now, 3)
      End If
*
      Return
*
      End
```

B. THRESHOLD MODEL

Most of what was written for the Invisible Kill case still applies. All variables retain their meanings. The variable "Threshold" is introduced to denote the optimal threshold, τ^o, which must be supplied to the simulation. In the current case optimal thresholds were computed by numerical solution of (3.10).

Fortran listing follows:

```
      PROGRAM Threshold Simulation
*
      REAL ProbAcq, TimeWindow, TimeAcq, Threshold, Rate
     +,Average, IntervalWidth, Variance, Lower, Upper
      INTEGER Iseed, NrMissiles, Replications, MissilesDown, k
*
      COMMON/ SHARE2/ MissilesDown
      COMMON/ SHARE3/ Iseed, ProbAcq, TimeAcq, Threshold, Rate,
     +NrMissiles
      COMMON/ SHARE4/ TimeWindow
      COMMON/ SHARE5/ Average, Variance, IntervalWidth, Upper, Lower
*
      OPEN( 15, FILE='/THRESH DATA')
      OPEN( 20, FILE='/THRESH OUTPUT')
*
      Read(15,*) Iseed
      Do k = 1, 16
*     Read input parameters and optimal threshold
        Read(15,*) Rate, NrMissiles, ProbAcq, TimeAcq, TimeWindow,
     +             Threshold
*
        Variance = 0.0
        Average = 0.0
        Replications = 0
        IntervalWidth = 1.0
*
*     Replicates at least 10,000 times and until 95% confidence interval is less than
*     or equal to 5% of the average.
        Do While( IntervalWidth .gt. 0.05*Average .or.
```

```
     +          Replications .le. 10000 )
           Replications = Replications + 1
           CALL INIT
           CALL SIMGO
           CALL STAT( MissilesDown, Replications)
         End Do
         Write(20,*) k, Lower, Average, Upper
       End Do
*
       CLOSE(15)
       CLOSE(20)
*
       Stop
*
       End
*
       SUBROUTINE ACQUISITION
*
       REAL X, Now, ProbAcq, TimeAcq, Threshold, Rate
       INTEGER Iseed, MissilesDown, NrMissiles
       LOGICAL Acquired
*
       COMMON/ SHARE1/ Now
       COMMON/ SHARE2/ MissilesDown
       COMMON/ SHARE3/ Iseed, ProbAcq, TimeAcq, Threshold, Rate,
      +NrMissiles
*
*      Sample from a Uniform(0,1) to determine outcome of acquisition.
       CALL RANNUM( 1, Iseed, 0.0, 1.0, 0, X)
       Acquired = .FALSE.
       If( X .le. ProbAcq ) Acquired = .TRUE.
       If( Acquired ) then
*       When missile is acquired schedule a "Shooting" event
         CALL SCHEDULE( Now+TimeAcq, 2)
       else
*       Upon failure to acquire schedule another acquisition in time = current time +
*       acquisition time + threshold.
         CALL SCHEDULE( Now+TimeAcq+Threshold, 1)
       End If
*
       Return
```

```
*
      End
*
      SUBROUTINE SHOOTING
*
      REAL Now, Tsample, Threshold, Rate, ProbAcq, TimeAcq
      INTEGER MissilesDown, Iseed, NrMissiles
      LOGICAL Destroyed
*
      COMMON/ SHARE1/ Now
      COMMON/ SHARE2/ MissilesDown
      COMMON/ SHARE3/ Iseed, ProbAcq, TimeAcq, Threshold, Rate,
     +NrMissiles
*
      SAVE SHARE2
*
*     Sample the exponential destruction time
*     CALL RANNUM( 3, Iseed, Rate, 0.0, 0.0, Tsample)
      Destroyed = .FALSE.
*     When the sampled time is less than the threshold time, the missile is killed
      If( Tsample .le. Threshold ) Destroyed = .TRUE.
      If( Destroyed ) then
        MissilesDown = MissilesDown + 1
*     If all missile have been shot ...
        If(MissilesDown .eq. NrMissiles) then
          Return
        else
*     If some missiles remain, schedule another acquisition immediately
          CALL SCHEDULE( Now+Tsample, 1)
        End If
      else
*     If the missile is not destroyed, schedule an acquisition in time = current time
*     + threshold time
        CALL SCHEDULE( Now+Threshold, 1)
      End If
*
      Return
*
      End
```

C. PERFECT TASK INFORMATION

We present programs for the simulation, "Perfect Task Info Simulation", and for the probability model, "Exact PMF for Perfect Task Info". This simulation, unlike the preceding, has no discrete event subroutines. It represents the case without the threshold improvement mentioned in Chapter V. The improved case, however, is simulated with minor modifications to the code presented here. The same applies to the plots displayed in Chapter IV, for the PMF of N(T). To generate those plots, we also used a small APL function which computes the frequency of the data, adapted from Thomson [Ref. 10].

The main program variables are:

- "E" is a real array for storage of the task times.
- "Sum" accumulates the expended time.
- "Time" is the time window, T.
- "Rate" is the rate parameter, λ, of the exponential task times.
- "p" is the probability of success in acquisition.
- "w" is the fixed setup or acquisition time.
- "S" is used to store samples from a Uniform distribution.
- "k", "kk", "kkk", are used as indices for "Do" loops.
- "n" is the number of missiles.

The commented simulation program is listed next. Subroutine SORT implements an insertion sorting algorithm, which closely follows the code presented by Etter [Ref. 11]. Subroutine STAT is also used by the previous simulations.

```
      PROGRAM Perfect Task Info Simulation
*
*     Simulation for the improved case
*
      REAL E(500), Sum, Time, Rate, p, w, S, Lower, Average, Upper,
     +Variance
      INTEGER Iseed, Replications, k, kk, kkk, n
*
      COMMON/ MAINSTAT/ Lower, Average, Upper, Variance
*
      OPEN( 10, FILE='/PACKGT DATA')
      OPEN( 15, FILE='/PACKGT OUTPUT')
*
      Read(10,*) Iseed, Replications
*
      Do kkk = 1, 16
        Read(10,*) Rate, n, p, w, Time
        Average= 0.0
        Variance = 0.0
*
        Do kk = 1, Replications
*         Sample the n task times from an Exponential
          Do k = 1, n
            CALL RANNUM( 3, Iseed, Rate, 0.0, 0.0, E(k) )
          End Do
*         Sorts the task times in ascending order of their lengths
          CALL SORT( n, E)
          k = 0
          Sum = 0.0
          Do While( k .le. n .and. Sum .lt. Time )
             k = k + 1
*            Sample from a Uniform to determine if acquisition successful
            CALL RANNUM( 1, Iseed, 0.0, 1.0, 0.0, S)
            If(S .le. p) then
*              If acquired pack one more job
```

```
            Sum = Sum + w + E(k)
          Else
*           If not acquired, expend time window
            Sum = Time + 1.0
          End If
        End Do
*       Update the statistics
      CALL STAT(k-1,kk)
      End Do
*
      Write(15,*) kkk, Lower, Average, Upper
    End Do
*
    CLOSE(10)
    CLOSE(15)
*
    Stop
*
    End
*
    SUBROUTINE SORT( n, E)
*
*   Sorts the elements of the array E in increasing order. E(1) = smallest;
*    E(n) = largest.
*
    INTEGER n, kk, k
    REAL E(n), Stor
    LOGICAL Done
*
    Do kk = 1, n-1
      If( E(kk) .gt. E(kk+1) ) then
        Done = .FALSE.
        k = kk
        Do While( .not. Done )
          Stor = E(k)
          E(k) = E(k+1)
          E(k+1) = Stor
          If( k .eq. 1 .or. E(k) .gt. E(k-1)) then
            Done = .TRUE.
          else
            k = k-1
```

```
          End If
        End Do
      End If
    End Do
*
    Return
*
    End
*
    SUBROUTINE STAT( y, n)
*
*   Computes sequential statistics
*
    INTEGER y, n
    REAL Average, Variance, Oldaverage, Upper, Intervalwidth,
   +Oldvariance, Lower, X
*
    COMMON/ MAINSTAT/ Lower, Average, Upper, Variance
*
    SAVE MAINSTAT
*
    Oldaverage = Average
    Oldvariance = Variance
    X = Real(y)
*
    If( n .eq. 1 ) then
      Average = X
      Variance = 0.0
      Intervalwidth = 1.0E20
    End If
*
    If(n .ge. 2) Average = (REAL(n-1)*Oldaverage+X)/Real(n)
    If(n .eq. 2) Variance = Oldaverage**2+(X**2)-(2.0*(Average**2))
    If(n .ge. 3) then
      Variance = (REAL(n-2)/REAL(n-1)) * Oldvariance +
   +   Oldaverage**2 - (REAL(n)/REAL(n-1)) * Average**2 +
   +   (X**2)/REAL(n-1)
    End If
    If(n .GE. 2) Intervalwidth = 2.0*1.96*SQRT(Variance)/SQRT(REAL(n))
    Lower = Average - Intervalwidth/2.0
    Upper = Average + Intervalwidth/2.0
```

```
*
      Return
*
      End
```

The program "Exact PMF for Perfect Task Info" implements the analytical results derived in Chapter IV. It should be run with option "(NOXUFLOW", to avoid underflow messages.

Additional variables in this program are:

- "PMF" - represents $P(N(T) = j)$.
- "FSj" - denotes $P(S_j > T)$.

The remaining variables retain their meanings and notation.

```
      PROGRAM Exact PMF for Perfect Task Info
*
      DOUBLE PRECISION Rate, Time, p, w, PMF, FSj, Average
      INTEGER j, n, k
*
      Open( 20, File = '/PEX DATA')
      Open( 25, File = '/PEX OUTPUT')
*
      Do k = 1, 16
        Read(20,*) Rate, n, p, w, Time
        Average = 0.0D0
        Do j = 0, n
*     Uses equation (4.23) to compute PMF
          PMF = (p**j)*(1.0D00-FSj(j,Time-DBLE(j)*w,n,Rate)) -
     +      (p**(j+1))*(1.0D00-FSj(j+1,Time-DBLE(j+1)*w,n,Rate))
*     Average is computed with (5.1)
          Average = Average + DBLE(j) * PMF
        End Do
        Write(25,*) k, Average
      End Do
```

```
      Close(20)
      Close(25)
*
      Stop
*
      End
*
      DOUBLE PRECISION FUNCTION FSj(jj,X,nn,Rrate)
*
*     Implements equation (4.17)
      INTEGER jj, i, k
      DOUBLE PRECISION X, Sum, Prod, Rrate, COMB, FACT
*
      If( X .le. 0 ) then
        FSj = 1.0D00
        Return
      End If
      If( jj .le. 0) then
        FSj = 0.0D00
      ElseIf((jj.ge.1) .and. (jj.le.(nn-1))) then
        Sum = 0.0D00
*     Compute the product term in equation (4.13)
        Do i = 1, jj
          Prod = 1.0D00
          If ( jj .ne. 1 ) then
            Do k = 1, jj
              If ( i .ne. k ) then
                Prod=Prod/(DBLE(nn-k+1)/DBLE(jj-k+1)-DBLE(nn-i+1)/
     +            DBLE(jj-i+1))
              End If
            End Do
          End If
          If(ABS(Rrate*(DBLE(nn-i+1)/DBLE(jj-i+1))*X) .lt. 170.0) then
            Sum=Sum+Prod*(DBLE(jj-i+1)/DBLE(nn-i+1))*DEXP(-Rrate*
     +       (DBLF(nn-i+1)/DBLE(jj-i+1))*X)
          End If
          FSj = Sum * COMB(nn,jj)
          If( ABS(FSj) .gt. 1.0D00) FSj = 1.0D00
        End Do
       ElseIf(jj .eq. nn) then
        Sum = 1.0D00
```

```
      Do k = 1, nn-1
        Sum = Sum + ((Rrate*X)**k)/FACT(k)
      End Do
      FSj = Sum * DEXP(-Rrate*X)
    ElseIf(jj .gt. nn) then
      FSj = 1.0D00
    End If
*
    Return
*
    End
*
    DOUBLE PRECISION FUNCTION COMB( m, p)
*
*   Calculates the binomial coefficients  (m p)
*
    INTEGER m, p, k
*
    COMB = 1.0D+00
*
    If( p .ne. m) then
      Do k = 1, p
        COMB = COMB*(DBLE(m-k+1)/DBLE(k))
      End Do
    End If
*
    Return
*
    End
*
    DOUBLE PRECISION FUNCTION FACT(X)
*
*    Returns the value of X!
    INTEGER X, K
*
    FACT = 1.0D+00
    If( X .ge. 2 ) then
      Do K = 2, X
        FACT = FACT * DBLE(K)
      End Do
    End If
```

```
*
      Return
*
      End
```

APPENDIX B. THRESHOLD MODEL GRAPHICS

This appendix contains plots for the Threshold model. The response surfaces shown were obtained in GRAFSTAT [Ref. 12], from simulation output.

In both plots the most salient feature is that the expected number of missiles killed is not very sensitive to the threshold **alone**.

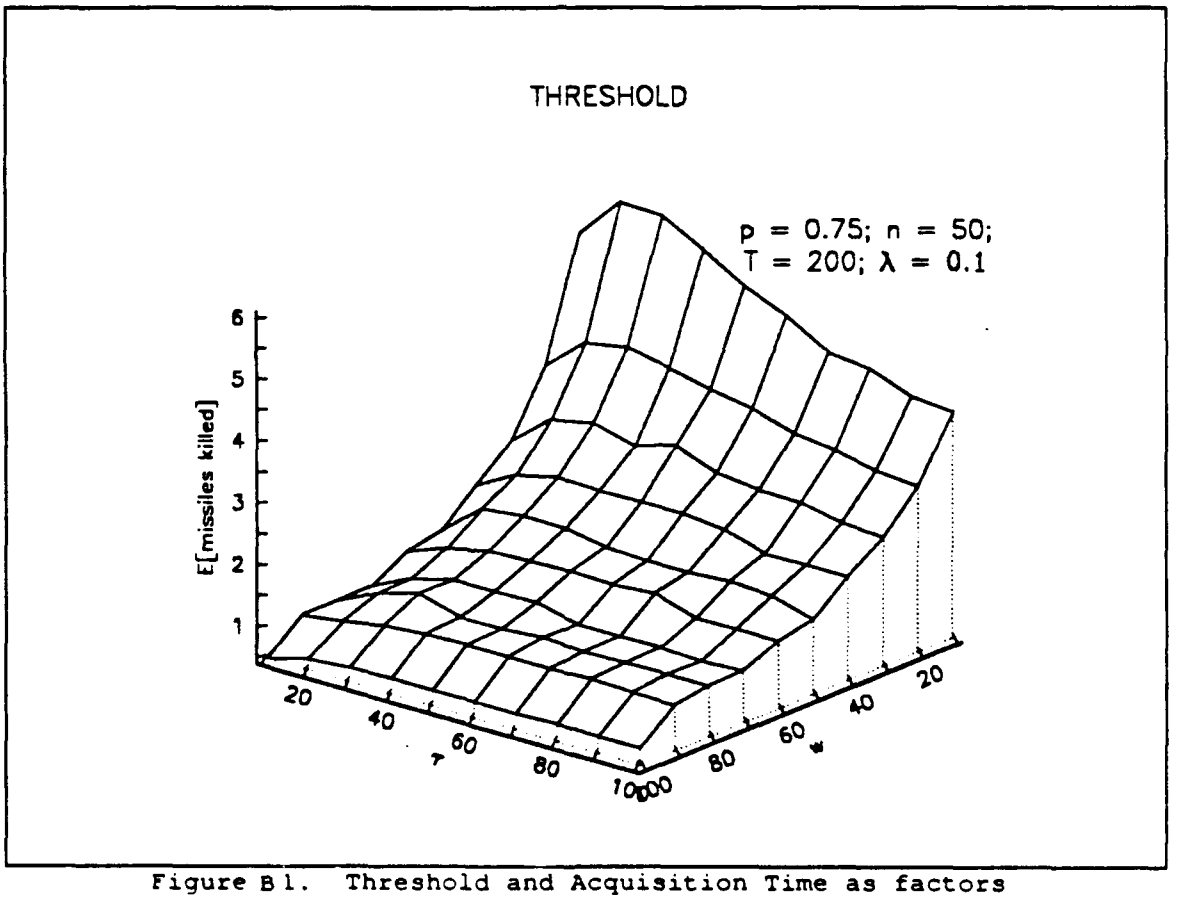

Figure B1. Threshold and Acquisition Time as factors

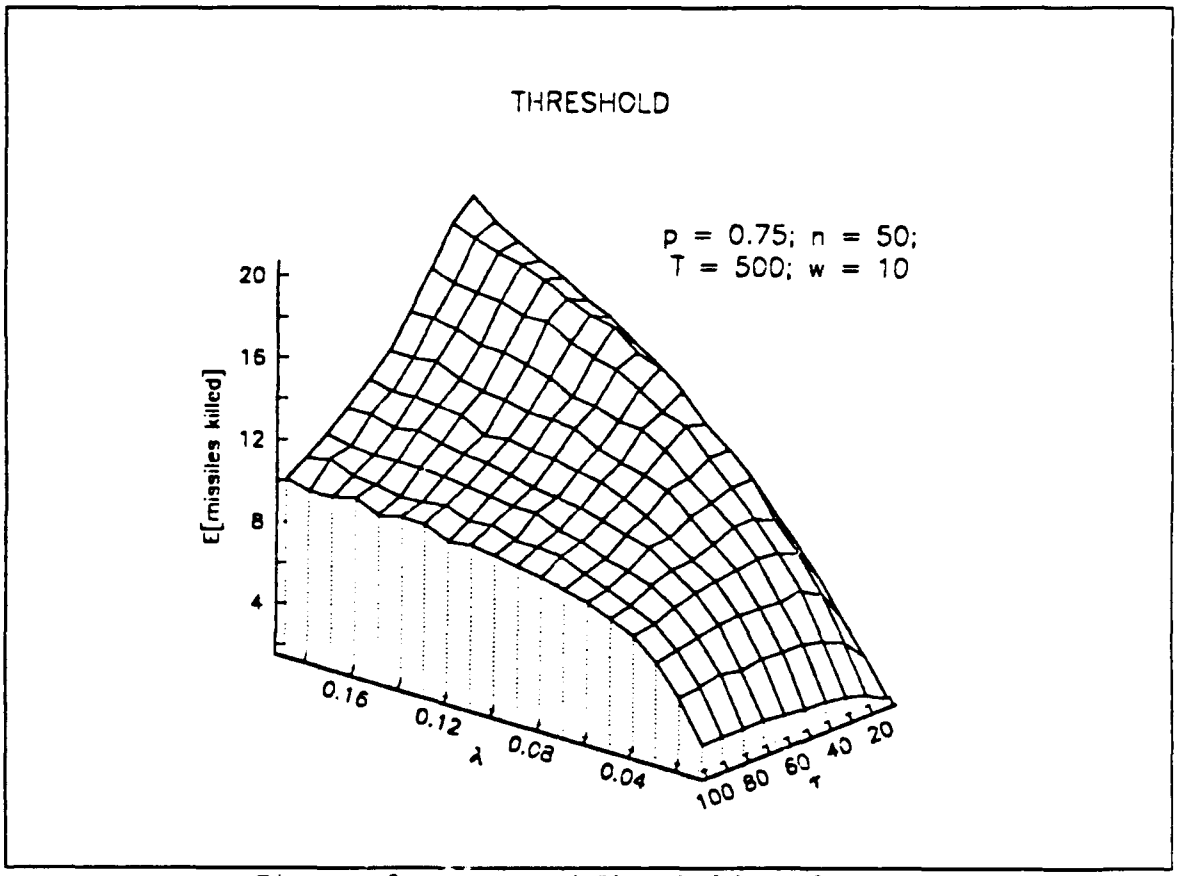

Figure B2. Rate and Threshold as factors

APPENDIX C. DISTRIBUTION OF S_j

In this appendix, we describe the evidence which suggested that a *Gamma* distribution would provide a suitable approximation to the distribution of S_j (the approximation is developed in subsection IV.C.1).

A slight adaptation of the program to simulate the packing problem (Perfect Task Information) was used, to produce samples of S_j for several values of j. These samples of size 1000, were used to construct Gamma probability plots. One such plot is depicted in Figure C1, for j=5.

Other plots, with different values of j, produced much similar results. As can be seen, the sampled distribution seems to closely follow the shape of a Gamma. In all plots slight departures from linearity were only observed at the right tail.

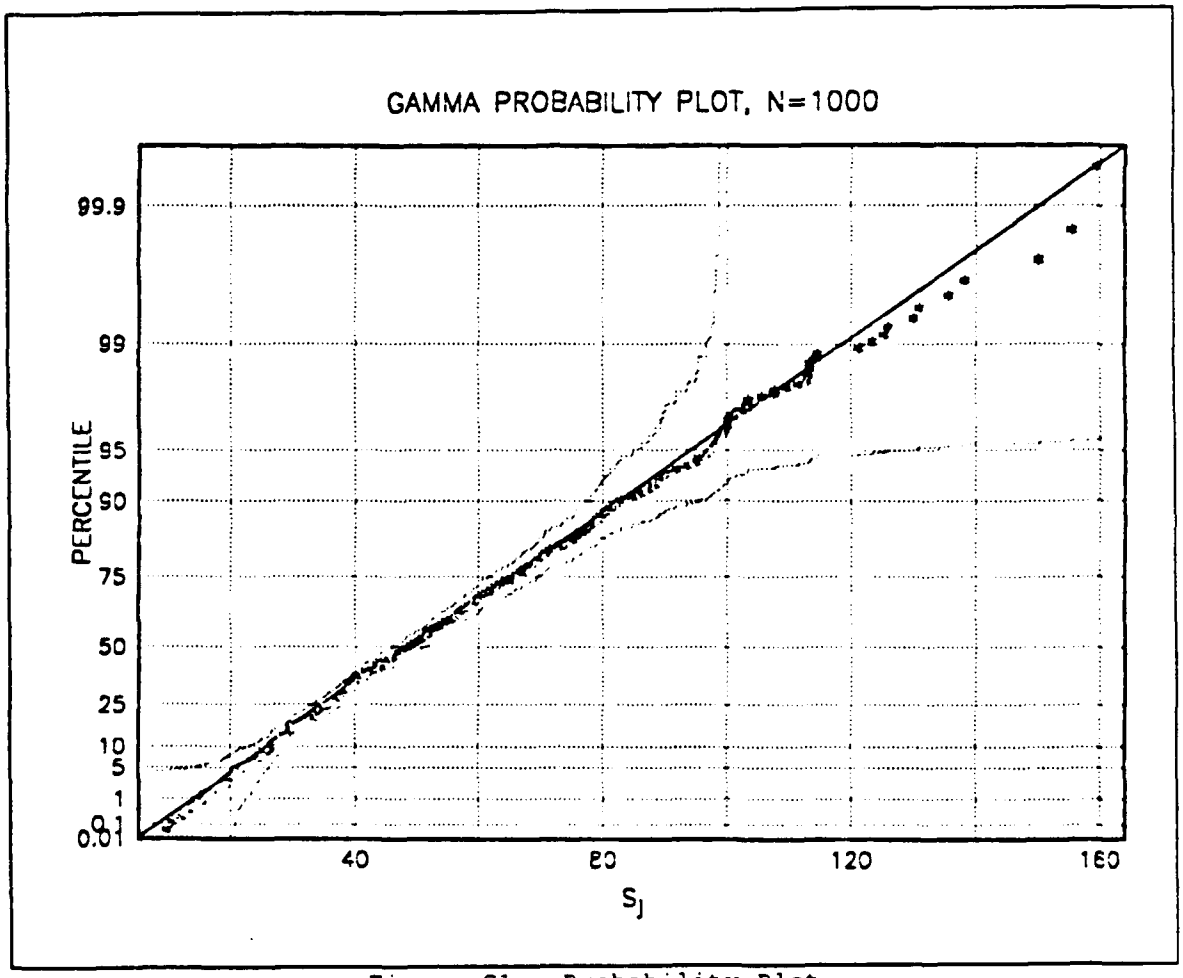

Figure C1. Probability Plot

LIST OF REFERENCES

1. Law, Averill M. and Kelton, W. David, *Simulation Modeling and Analysis*, pp. 280-281, McGraw-Hill, 1982.

2. Bailey, M. P., *SIMUTIL FORTRAN Simulation Utility Subroutines*, Unpublished Lecture Notes, System Simulation, Department of Operations Research, Naval Postgraduate School, Monterey, CA 93943-5000, 1990.

3. Lewis, P. A., and Uribe, L., *The New Naval Postgraduate School Random Number Package - LLRANDOMII*, Naval Postgraduate School Technical Report, NPS-55-81-005, 1981.

4. Taylor, Howard M. and Karlin, Samuel, *An Introduction to Stochastic Modeling*, p. 289, Academic Press, 1984.

5. Coffman Jr., E. G., Fayolle G., Jacquet P., and Robert, P.,"Largest-First Sequential Selection with a Sum Constraint", *Operations Research Letters*, v. 9, pp. 141-146, May 1990.

6. Coffman Jr., E. G., Flatto L. and Weber R. R., "Optimal Selection of Stochastic Intervals Under a Sum Constraint", *Advances in Applied Probability*, v. 19, pp. 454-473, 1987.

7. Barlow, Richard E., Proschan, Frank, *Statistical Theory of Reliability and Life Testing - Probability Models*, Chapter 3, pp. 59-60, To Begin With, 1981.

8. Feller, *An Introduction to Probability Theory and its Applications*, v. II, p. 40, John Wiley & Sons, 1971.

9. Box, George E. P., Hunter, William G., and Hunter, J. Stuart, *Statistics for Experimenters*, Chapter 12, John Wiley & Sons, 1978.

10. Thomson, Norman, *APL Programs for the Mathematics Classroom*, pp. 109, Springer-Verlag, 1989.

11. Etter, D. M., *Structured Fortran 77 for Engineers and Scientists*, pp. 210-212, Benjamin/Cummings, 1990.

12. Stein, D. M., van der Hoeven, W., Welch, P. D., *GRAFSTAT Primer*, IBM Research Yorktown Heights, N.Y., September 1987.

INITIAL DISTRIBUTION LIST

		No.Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Rui Almeida Est. Moscavide, 58, 1º Dto. 1800 Olivais, Lisboa Portugal	2
4.	Prof. Donald P. Gaver, Code OR/Gv Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	4
5.	Prof. Patricia A. Jacobs, Code OR/Jc Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Centro de Investigação Operacional da Armada Edifício da Administração Central de Marinha Praça do Comércio, 1100 Lisboa Portugal	2
7.	Direcção do Serviço de Instrução e Treino Edifício da Administração Central de Marinha Praça do Comércio, 1100 Lisboa Portugal	2